19.12.03

日本国特許庁
JAPAN PATENT OFFICE

別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office.

出願年月日
Date of Application: ２００２年１２月１９日

出願番号
Application Number: 特願２００２－３６７６０８

[ST. 10/C]: [ＪＰ２００２－３６７６０８]

出願人
Applicant(s): エヌ・ティ・ティ・コミュニケーションズ株式会社

２００４年　１月２９日

特許庁長官
Commissioner,
Japan Patent Office

今井康夫

出証番号　出証特２００４－３００３８５３

【書類名】　特許願
【整理番号】　GLN-00388
【提出日】　平成14年12月19日
【あて先】　特許庁長官殿
【国際特許分類】　G06F 17/00
【発明の名称】　データ分割方法および装置
【請求項の数】　2
【発明者】
【住所又は居所】　東京都千代田区内幸町一丁目１番６号　エヌ・ティ・ティ・コミュニケーションズ株式会社内
【氏名】　荻原　利彦
【発明者】
【住所又は居所】　東京都千代田区内幸町一丁目１番６号　エヌ・ティ・ティ・コミュニケーションズ株式会社内
【氏名】　野村　進
【特許出願人】
【識別番号】　399035766
【氏名又は名称】　エヌ・ティ・ティ・コミュニケーションズ株式会社
【代理人】
【識別番号】　100083806
【弁理士】
【氏名又は名称】　三好　秀和
【電話番号】　03-3504-3075
【選任した代理人】
【識別番号】　100068342
【弁理士】
【氏名又は名称】　三好　保男

【選任した代理人】

【識別番号】　100095500

【弁理士】

【氏名又は名称】　伊藤　正和

【選任した代理人】

【識別番号】　100101247

【弁理士】

【氏名又は名称】　高橋　俊一

【選任した代理人】

【識別番号】　100098327

【弁理士】

【氏名又は名称】　高松　俊雄

【手数料の表示】

【予納台帳番号】　001982

【納付金額】　21,000円

【提出物件の目録】

【物件名】　明細書　1

【物件名】　図面　1

【物件名】　要約書　1

【包括委任状番号】　9908855

【プルーフの要否】　要

【書類名】　明細書

【発明の名称】　データ分割方法および装置

【特許請求の範囲】

【請求項１】　元データを所望の処理単位ビット長に基づいて所望の分割数の分割データに分割するデータ分割方法であって、

元データを処理単位ビット長毎に区分けして、分割数より１つ少ない複数の元部分データを生成し、

この複数の元部分データの各々に対応して処理単位ビット長の複数の乱数部分データを生成し、

各分割データを処理単位ビット長毎に区分けして分割数より１つ少ない複数の分割部分データを生成した場合の各分割部分データを元部分データと乱数部分データの排他的論理和を含む所定の定義式に従って生成すること

を特徴とするデータ分割方法。

【請求項２】　元データを所望の処理単位ビット長に基づいて所望の分割数の分割データに分割するデータ分割装置であって、

元データを処理単位ビット長毎に区分けして、分割数より１つ少ない複数の元部分データを生成する元部分データ生成手段と、

この複数の元部分データの各々に対応して処理単位ビット長の複数の乱数部分データを生成する乱数生成手段と、

各分割データを処理単位ビット長毎に区分けして分割数より１つ少ない複数の分割部分データを生成した場合の各分割部分データを元部分データと乱数部分データの排他的論理和を含む所定の定義式に従って生成する分割部分データ生成手段と

を有することを特徴とするデータ分割装置。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】

本発明は、データの機密性および安全性を確保するためにはデータを分割して保管することが有効であるが、このような場合などに有効なデータ分割方法およ

び装置に関し、更に詳しくは、元データを所望の処理単位ビットに基づいて所望の分割数の分割データに分割するデータ分割方法および装置に関する。

【０００２】

【従来の技術】

重要な秘密データ（以下、元データという）を保管する場合、紛失、破壊、盗難やプライバシー侵害の脅威がある。このような脅威は秘密に保管すべきデータを単に暗号化しただけでは解決できず、紛失、破壊に備えてコピーを複数作ることが有効であるが、コピーを複数作ると盗難のリスクが増加してしまう。

【０００３】

このような問題を解決する手段として、従来、しきい値秘密分散法がある。この従来の方法は、元データSをn個のデータに分割し、そのうち任意のx個の分割データを集めれば元データSが復元できるが、任意のx-1個の分割データでは元データSは復元できないというものである。従って、x-1個まで分割データが盗まれても元データSが漏れず、またn-x個まで分割データを紛失したり破壊されたりしても、元データSを復元できる。

【０００４】

この方法の代表的な実現例としてm-1次多項式と剰余演算により構成される方法がある（非特許文献２参照）。この従来の方法は、公開鍵暗号方式の秘密鍵の分割管理などで利用されており、データ量が１キロバイトに満たない程度であるため、現状のコンピュータの演算処理能力、記憶装置・記憶媒体などのコストに対しては特に問題ない。

【０００５】

【非特許文献１】

Shamir,A"How to Share a Secret"Comm.Assoc.Comput.Mach.,vol.22,no.11,pp.612-613(Nov.1979)

【非特許文献２】

Bruce Schneier"Applied Cryptography",John Wiley & Sons,Inc.,pp,383-384(1994)

【０００６】

【発明が解決しようとする課題】

上述した従来の方法を安全に保管したいデータ量が例えばメガバイト、ギガバイトまたはそれ以上の規模となった場合に利用すると、多項式演算・剰余演算などを含む多倍長整数の演算処理を大量のデータに対して行う演算処理能力が必要となるとともに、またこの従来の方法では、例えば分割数n=5の場合には１バイトのデータから１バイトの分割データが５つ生成されるため、元データに対して単純に分割数に比例した倍数の記憶容量が必要となるなど、コンピュータを用いて具体的に実現する上で現実的ではないという問題がある。

【０００７】

また、上述した従来の方法では、データの機密性を確保するため分割演算のための処理単位を設定しているが、この分割演算の処理単位にある程度のデータ長が必要となり、任意の処理単位で分割演算を行うことができないという問題もある。

【０００８】

本発明は、上記に鑑みてなされたもので、その目的とするところは、比較的簡単な処理により元データを効率的に分割し得るデータ分割方法および装置を提供することにある。

【０００９】

【課題を解決するための手段】

上記目的を達成するため、請求項１記載の本発明は、元データを所望の処理単位ビット長に基づいて所望の分割数の分割データに分割するデータ分割方法であって、元データを処理単位ビット長毎に区分けして、分割数より１つ少ない複数の元部分データを生成し、この複数の元部分データの各々に対応して処理単位ビット長の複数の乱数部分データを生成し、各分割データを処理単位ビット長毎に区分けして分割数より１つ少ない複数の分割部分データを生成した場合の各分割部分データを元部分データと乱数部分データの排他的論理和を含む所定の定義式に従って生成することを要旨とする。

【００１０】

請求項１記載の本発明にあっては、元データを処理単位ビット長毎に区分けし

て複数の元部分データを生成し、複数の乱数部分データを生成し、各分割データを構成する各分割部分データを元部分データと乱数部分データの排他的論理和からなる所定の定義式に従って生成するため、従来のように多項式や剰余演算を用いることなく、コンピュータ処理に適したビット演算である排他的論理和演算を用いることにより高速かつ高性能な演算処理能力を必要とせず、大容量のデータに対しても簡単な演算処理を繰り返して分割データを簡単かつ迅速に生成することができるとともに、また分割データの保管に必要となる記憶容量も分割数に比例した倍数の容量よりも小さくすることができる。

【００１１】

また、前記所定の定義式が、元データ、乱数、分割データ、分割数および処理単位ビット長をそれぞれS,R,D,nおよびbで表すとともに、複数n個のうちの１つを表わす変数としてi(=1～n)およびj(=1～n-1)を用いて複数(n-1)個の元部分データ、複数(n-1)個の乱数部分データ、複数(n)個の分割データおよび各分割データの複数(n-1)個の分割部分データのそれぞれのうちの１つをそれぞれS(i),R(i),D(i)およびD(i,j)で表わし、変数iを１からn-1まで変えて、各元部分データS(i)を元データSのb×(i-1)+1ビット目からbビット分のデータとして作成し、U[n,n]をn×n行列である上三角行列とし、P[n,n]をn×n行列である回転行列としたとき、c(j,i,k)を(n-1)×(n-1)行列であるU[n-1,n-1]×(P[n-1,n-1])^(j-1)のi行k列の値と定義し、c(j,i,k)=1のとき、Q(j,i,k)=R(k),c(j,i,k)=0のとき、Q(j,i,k)=0と定義したとき、変数iを１からnまで変えながら、各変数iにおいて変数jを１からn-1まで変えた場合において、i<nのとき、各分割部分データD(i,j)を

【数１】

$$D(i,j) = S(j) * \left(\prod_{k=1}^{n-1} Q(j,i,k) \right)$$

と設定し、i=nのとき、各分割部分データD(i,j)を

$$D(i,j)=R(j)$$

と設定し、上記操作を元データSの先頭から末尾まで繰り返し行うことにより分割数nの分割データを生成することを要旨としても良い。

【００１２】

この場合にあっては、所定の定義式が元部分データと乱数部分データの排他的論理和からなるため、従来のように多項式や剰余演算を行う高速かつ高性能な演算処理能力を必要とせず、大容量のデータに対しても簡単な演算処理を繰り返して分割データを簡単かつ迅速に生成することができる。

【００１３】

また、請求項２記載の本発明は、元データを所望の処理単位ビット長に基づいて所望の分割数の分割データに分割するデータ分割装置であって、元データを処理単位ビット長毎に区分けして、分割数より１つ少ない複数の元部分データを生成する元部分データ生成手段と、この複数の元部分データの各々に対応して処理単位ビット長の複数の乱数部分データを生成する乱数生成手段と、各分割データを処理単位ビット長毎に区分けして分割数より１つ少ない複数の分割部分データを生成した場合の各分割部分データを元部分データと乱数部分データの排他的論理和を含む所定の定義式に従って生成する分割部分データ生成手段とを有することを要旨とする。

【００１４】

請求項２記載の本発明にあっては、元データを処理単位ビット長毎に区分けして複数の元部分データを生成し、複数の乱数部分データを生成し、各分割データを構成する各分割部分データを元部分データと乱数部分データの排他的論理和からなる所定の定義式に従って生成するため、従来のように多項式や剰余演算を用いることなく、コンピュータ処理に適したビット演算である排他的論理和演算を用いることにより高速かつ高性能な演算処理能力を必要とせず、大容量のデータに対しても簡単な演算処理を繰り返して分割データを簡単かつ迅速に生成することができるとともに、また分割データの保管に必要となる記憶容量も分割数に比例した倍数の容量よりも小さくすることができる。

【００１５】

また、前記所定の定義式が、元データ、乱数、分割データ、分割数および処理単位ビット長をそれぞれS,R,D,nおよびbで表すとともに、複数n個のうちの１つを表わす変数としてi(=1～n)およびj(=1～n-1)を用いて複数(n-1)個の元部分データ、複数(n-1)個の乱数部分データ、複数(n)個の分割データおよび各分割デー

タの複数(n-1)個の分割部分データのそれぞれのうちの１つをそれぞれS(j),R(j),D(i)およびD(i,j)で表わし、変数iを１からn-1まで変えて、各元部分データS(j)を元データSのb×(j-1)+1ビット目からbビット分のデータとして作成し、U[n,n]をn×n行列である上三角行列とし、P[n,n]をn×n行列である回転行列としたとき、c(j,i,k)を(n-1)×(n-1)行列であるU[n-1,n-1]×(P[n-1,n-1])^(j-1)のi行k列の値と定義し、c(j,i,k)=1のとき、Q(j,i,k)=R(k),c(j,i,k)=0のとき、Q(j,i,k)=0と定義したとき、変数iを１からnまで変えながら、各変数iにおいて変数jを１からn-1まで変えた場合において、i<nのとき、各分割部分データD(i,j)を

【数２】

$$D(i,j) = S(j)*\left(\prod_{k=1}^{n-1} Q(j,i,k)\right)$$

と設定し、i=nのとき、各分割部分データD(i,j)を

D(i,j)=R(j)

と設定し、上記操作を元データSの先頭から末尾まで繰り返し行うことにより分割数nの分割データを生成することを要旨とする。

【００１６】

この場合にあっては、所定の定義式が元部分データと乱数部分データの排他的論理和からなるため、従来のように多項式や剰余演算を行う高速かつ高性能な演算処理能力を必要とせず、大容量のデータに対しても簡単な演算処理を繰り返して分割データを簡単かつ迅速に生成することができる。

【００１７】

【発明の実施の形態】

以下、図面を用いて本発明の実施の形態を説明する。図１は、本発明の一実施形態に係るデータ分割方法を実施するデータ分割装置を含むシステム構成図である。本実施形態のデータ分割装置は、符号１で示すように分割装置１としてネットワーク３に接続されて設けられ、このネットワーク３にアクセスしてくる端末５からの元データ分割要求に応じて元データSを複数の分割データに分割し、この分割した複数の分割データをネットワーク３を介して複数の保管サーバ７ａ，７ｂ，７ｃに保管するようになっている。なお、図１では、分割装置１は、端末

５からの元データSを３つの分割データD(1),D(2),D(3)に分割し、それぞれを複数の保管サーバ７ａ，７ｂ，７ｃに保管するようにしている。

【００１８】

また、分割装置１は、ネットワーク３を介してアクセスしてくる端末５からの元データ復元要求に応じて複数の分割データD(1),D(2),D(3)をネットワーク３を介して各保管サーバ７から取得し、この取得した複数の分割データD(1),D(2),D(3)から元データSを復元し、ネットワーク３を介して端末５に送信するようになっている。

【００１９】

分割装置１は、詳しくは、元データSから複数の分割データDを生成する分割データ生成手段１１、複数の分割データDから元データSを復元する元データ復元手段１３、元データSから複数の分割データDを生成するために使用される乱数Rを発生する乱数発生手段１５、および分割データ生成手段１１で生成した複数の分割データDをネットワーク３を介して複数の保管サーバ７ａ，７ｂ，７ｃに送信したり、また複数の保管サーバ７ａ，７ｂ，７ｃからの複数の分割データDをネットワーク３を介して受信するためのデータ送受信手段１７から構成されている。

【００２０】

上述したように構成される本実施形態における元データの分割および復元では、元データを所望の処理単位ビット長に基づいて所望の分割数の分割データに分割するが、この場合の処理単位ビット長は任意の値に設定することができ、元データを処理単位ビット長毎に区分けして、この元部分データから分割部分データを分割数より１少ない数ずつ生成するので、元データのビット長が処理単位ビット長の（分割数-1）倍の整数倍に一致しない場合は、元データの末尾の部分に０を埋めるなどして元データのビット長を処理単位ビット長の（分割数-1）倍の整数倍に合わせることにより本実施形態を適用することができる。

【００２１】

また、上述した乱数も（分割数-1）個の元部分データの各々に対応して処理単位ビット長のビット長を有する（分割数-1）個の乱数部分データとして乱数発生

手段１５から生成される。すなわち、乱数は処理単位ビット長毎に区分けされて、処理単位ビット長のビット長を有する（分割数-1）個の乱数部分データとして生成される。更に、元データは処理単位ビット長に基づいて所望の分割数の分割データに分割されるが、この分割データの各々も（分割数-1）個の元部分データの各々に対応して処理単位ビット長のビット長を有する（分割数-1）個の分割部分データとして生成される。すなわち、分割データの各々は、処理単位ビット長毎に区分けされて、処理単位ビット長のビット長を有する（分割数-1）個の分割部分データとして生成される。

【００２２】

なお、以下の説明では、上述した元データ、乱数、分割データ、分割数および処理単位ビット長をそれぞれS,R,D,nおよびbで表すとともに、また複数のデータや乱数などのうちの１つを表わす変数としてi(=1～n)およびj(=1～n-1)を用い、（分割数n-1）個の元部分データ、（分割数n-1）個の乱数部分データ、および分割数n個の分割データDのそれぞれのうちの1つをそれぞれS(j),R(j)およびD(i)で表記し、更に各分割データD(i)を構成する複数(n-1)の分割部分データをD(i,j)で表記するものとする。すなわち、S(j)は、元データSの先頭から処理単位ビット長毎に区分けして１番から順に採番した時のj番目の元部分データを表すものである。

【００２３】

この表記を用いると、元データ、乱数データ、分割データとこれらをそれぞれ構成する元部分データ、乱数部分データ、分割部分データは、次のように表記される。

【００２４】

【数３】

元データS=(n-1)個の元部分データS(j)

$$=S(1),S(2),\cdots,S(n-1)$$

乱数R=(n-1)個の乱数部分データR(j)

$$=R(1),R(2),\cdots,R(n-1)$$

n個の分割データD(i)=D(1),D(2),…,D(n)

各分割部分データD(i,j)

$$=\begin{matrix} D(1,1), D(1,2), \cdots, D(1,n-1) \\ D(2,1), D(2,2), \cdots, D(2,n-1) \\ \cdots \quad \cdots \quad \cdots \\ D(n,1), D(n,2), \cdots, D(n,n-1) \end{matrix}$$

$$(i=1\sim n),\ (j=1\sim n-1)$$

本実施形態は、上述したように処理単位ビット長毎に区分けされる複数の部分データに対して元部分データと乱数部分データの排他的論理和演算（XOR）を行って、詳しくは、元部分データと乱数部分データの排他的論理和演算（XOR）からなる定義式を用いて、元データの分割を行うことを特徴とするものであり、上述したデータ分割処理に多項式や剰余演算を用いる従来の方法に比較して、コンピュータ処理に適したビット演算である排他的論理和（XOR）演算を用いることにより高速かつ高性能な演算処理能力を必要とせず、大容量のデータに対しても簡単な演算処理を繰り返して分割データを生成することができるとともに、また分割データの保管に必要となる記憶容量も分割数に比例した倍数の容量よりも小さくすることができる。更に、任意に定めた一定の長さ毎にデータの先頭から順に演算処理を行うストリーム処理により分割データが生成される。

【００２５】

なお、本実施形態で使用する排他的論理和演算（XOR）は、以下の説明では、「＊」なる演算記号で表すことにするが、この排他的論理和演算のビット毎の演算規則での各演算結果は下記のとおりである。

【００２６】

0 * 0 の演算結果は 0

0 * 1 の演算結果は 1

1 * 0 の演算結果は 1

1 * 1 の演算結果は 0

また、XOR演算は交換法則、結合法則が成り立つ。すなわち、

a*b=b*a

(a*b)*c=a*(b*c)

が成り立つことが数学的に証明される。

【００２７】

また、a*a=0, a*0=0*a=aが成り立つ。

【００２８】

ここでa, b, cは同じ長さのビット列を表し、0はこれらと同じ長さですべて「0」からなるビット列を表す。

【００２９】

次に、フローチャートなどの図面も参照して、上記実施形態の作用について説明するが、この説明の前に図２、図５、図８、図９のフローチャートに示す記号の定義について説明する。

【００３０】

（１）$\Pi_{i=1}^{n}A(i)$は、A(1)*A(2)*…A(n)を意味するものとする。

【００３１】

（２）c(j, i, k)を(n-1)×(n-1)行列である。U[n-1, n-1]×(P[n-1, n-1])^(j-1)のi行k列の値と定義する。

【００３２】

このときQ(j, i, k)を下記のように定義する。

【００３３】

c(j, i, k)=1 のとき Q(j, i, k)=R((n-1)×m+k)

c(j, i, k)=0 のとき Q(j, i, k)=0

（３）U[n, n]とは、n×n行列であって、i行j列の値をu(i, j)で表すと、

i+j<=n+1 のとき u(i, j)=1

i+j>n+1 のとき u(i, j)=0

である行列を意味するものとし、「上三角行列」ということとする。具体的には下記のような行列である。

【００３４】

【数４】

$$U[3,3]=\begin{pmatrix}1&1&1\\1&1&0\\1&0&0\end{pmatrix}\qquad U[4,4]=\begin{pmatrix}1&1&1&1\\1&1&1&0\\1&1&0&0\\1&0&0&0\end{pmatrix}$$

（４）P[n,n]とは、n×n行列であって、i行j列の値をp(i,j)で表すと、

j=i+1　　のとき p(i,j)=1

i=1,j=n　のとき p(i,j)=1

上記以外のとき　p(i,j)=0

である行列を意味するものとし、「回転行列」ということとする。具体的には下記のような行列であり、他の行列の右側からかけると当該他の行列の１列目を２列目へ、２列目を３列目へ、…,n-1列目をn列目へ、n列目を１列目へ移動させる作用がある。つまり、行列Pを他の行列に右側から複数回かけると、その回数分だけ各列を右方向へ回転させるように移動させることができる。

【００３５】

【数５】

$$P[3,3]=\begin{pmatrix}0&1&0\\0&0&1\\1&0&0\end{pmatrix}\qquad U[4,4]=\begin{pmatrix}0&1&0&0\\0&0&1&0\\0&0&0&1\\1&0&0&0\end{pmatrix}$$

（５）A,Bをn×n行列とすると、A×Bとは行列AとBの積を意味するものとする。行列の成分同士の計算規則は通常の数学で用いるものと同じである。

【００３６】

（６）Aをn×n行列とし、iを整数とすると、A＾iとは行列Aのi個の積を意味するものとする。また、A＾0とは単位行列Eを意味するものとする。

【００３７】

（７）単位行列E[n,n]とは、n×n行列であって、i行j列の値をe(i,j)で表すと

i=j　　のとき e(i,j)=1

上記以外のとき e(i,j)=0

である行列を意味するものとする。具体的には下記のような行列である。Aを任意のn×n行列とすると

A×E=E×A=A

となる性質がある。

【００３８】

【数６】

$$E[3,3]=\begin{pmatrix}1&0&0\\0&1&0\\0&0&1\end{pmatrix}\qquad E[4,4]=\begin{pmatrix}1&0&0&0\\0&1&0&0\\0&0&1&0\\0&0&0&1\end{pmatrix}$$

次に、図２に示すフローチャートおよび図３、図４に示す具体的データなどを参照して、上記実施形態の作用として、まず元データSの分割処理について説明する。

【００３９】

本実施形態の分割装置１の利用者は、端末５からネットワーク３を介して分割装置１にアクセスし、分割装置１に元データSを送信し、分割装置１ではデータ送受信手段１７が端末５からの元データSを受信し、分割装置１に供給する（図２のステップＳ２０１）。なお、本例では、元データSは、１６ビットの「10110010 00110111」とする。

【００４０】

次に、利用者は端末５から分割数nとして３を分割装置１に指示する（ステップＳ２０３）。この分割数nは分割装置１において予め定められた値を用いてもよい。なお、この分割数n=3に従って分割装置１で生成される３個の分割データをD(1),D(2),D(3)とする。この分割データD(1),D(2),D(3)は、すべて元データのビット長と同じ１６ビット長のデータである。

【００４１】

それから、元データSを分割するために使用される処理単位ビット長bを８ビットと決定し、また元データと同じビット長である１６ビットの乱数Rを乱数発生手段１５から取得して生成する（ステップＳ２０５）。この処理単位ビット長bは、利用者が端末５から分割装置１に対して指定してもよいし、または分割装置１において予め定められた値を用いてもよい。なお、処理単位ビット長bは、任意のビット数でよいが、ここでは元データSを割り切れることができる８ビットとしている。従って、上記１６ビットの「10110010 00110111」の元データSは、８ビットの処理単位ビット長で区分けされた場合の２個の元分割データS(1)およびS(2)は、それぞれ「10110010」および「00110111」となる。

【００４２】

次のステップＳ２０７では、元データSのビット長が8×2の整数倍であるか否かを判定し、整数倍でない場合には、元データSの末尾を０で埋めて、8×2の整数倍に合わせる。なお、本例のように処理単位ビット長bが８ビットおよび分割数nが３に設定された場合における分割処理は、元データSのビット長として１６ビットに限られるものでなく、処理単位ビット長b×(分割数n-1)=8×2の整数倍の元データSに対して有効なものである。

【００４３】

次に、ステップＳ２０９では、変数m、すなわち上述した整数倍を意味する変数mを０に設定する。本例のように、元データSが処理単位ビット長b×(分割数n-1)=8×2=16ビットである場合には、変数mは０であるが、２倍の３２ビットの場合には、変数mは１となり、３倍の４８ビットの場合には、変数mは２となる。

【００４４】

次に、元データSの8×2×m+1ビット目から8×2ビット分のデータが存在するか否かが判定される（ステップＳ２１１）。これは、このステップＳ２１１以降に示す分割処理を元データSの変数mで特定される処理単位ビット長b×(分割数n-1)=8×2=16ビットに対して行った後、元データSとして次の１６ビットがあるか否かを判定しているものである。本例のように元データSが１６ビットである場合には、１６ビットの元データSに対してステップＳ２１１以降の分割処理を１回

行うと、後述するステップＳ２１９で変数mが+1されるが、本例の元データSでは変数mがm+1の場合に相当する１７ビット以降のデータは存在しないので、ステップＳ２１１からステップＳ２２１に進むことになるが、今の場合は、変数mは０であるので、元データSの8×2×m+1ビット目は、8×2×0+1=1となり、元データSの１６ビットの１ビット目から8×2ビット分にデータが存在するため、ステップＳ２１３に進む。

【００４５】

ステップＳ２１３では、変数jを１から２(=分割数n-1)まで変えて、元データSの8×(2×m+j-1)+1ビット目から８ビット分(=処理単位ビット長)のデータを元部分データS(2×m+j)に設定し、これにより元データSを処理単位ビット長で区分けした２(分割数n-1)個の元部分データS(1),S(2)を次のように生成する。

【００４６】

元データS=S(1),S(2)

第１の元部分データS(1)=「10110010」

第２の元部分データS(2)=「00110111」

次に、変数jを１から２(=分割数n-1)まで変えて、乱数部分データR(2×m+j)に乱数発生手段１５から発生する８ビットの長さの乱数を設定し、これにより乱数Rを処理単位ビット長で区分けした２(分割数n-1)個の乱数部分データR(1),R(2)を次のように生成する（ステップＳ２１５）。

【００４７】

乱数R=R(1),R(2)

第１の乱数部分データR(1)=「10110001」

第２の乱数部分データR(2)=「00110101」

次に、ステップＳ２１７において、変数iを１から３(=分割数n)まで変えるとともに、更に各変数iにおいて変数jを１から２(=分割数n-1)まで変えながら、ステップＳ２１７に示す分割データを生成するための元部分データと乱数部分データの排他的論理和からなる定義式により複数の分割データD(i)の各々を構成する各分割部分データD(i,2×m+j)を生成する。この結果、次に示すような分割データDが生成される。

【００４８】

【数７】

分割データD

＝３個の分割データD(i)=D(1),D(2),D(3)

第１の分割データD(1)

＝２個の分割部分データD(1,j)=D(1,1),D(1,2)

＝「00110110」，「10110011」

第２の分割データD(2)

＝２個の分割部分データD(2,j)=D(2,1),D(2,2)

＝「00000011」，「00000010」

第３の分割データD(3)

＝２個の分割部分データD(3,j)=D(3,1),D(3,2)

＝「10110001」，「00110101」

なお、各分割部分データ(i,j)を生成するためのステップＳ２１７に示す定義式は、本例のように分割数n=3の場合には、具体的には図４に示す表に記載されているものとなる。図４に示す表から、分割部分データD(1,1)を生成するための定義式はS(1)*R(1)*R(2)であり、D(1,2)の定義式はS(2)*R(1)*R(2)であり、D(2,1)の定義式はS(1)*R(1)であり、D(2,2)の定義式はS(2)*R(2)であり、D(3,1)の定義式はR(1)であり、D(3,2)の定義式はR(2)である。また、図４に示す表にはm>0の場合の任意の整数についての一般的な定義式も記載されている。

【００４９】

このように整数倍を意味する変数m=0の場合について分割データDを生成した後、次に変数mを１増やし（ステップＳ２１９）、ステップＳ２１１に戻り、変数m+1に該当する元データSの１７ビット以降について同様の分割処理を行おうとするが、本例の元データSは１６ビットであり、１７ビット以降のデータは存在しないので、ステップＳ２１１からステップＳ２２１に進み、上述したように生成した分割データD(1),D(2),D(3)を分割装置１のデータ送受信手段１７からネットワーク３を介して保管サーバ７ａ，７ｂ，７ｃにそれぞれ送信し、各保管サーバ７に保管し、分割処理を終了する。なお、このように保管された分割データD(1)

,D(2),D(3)はそれぞれ単独では元データが推測できない。

【００５０】

ここで、上述した図２のフローチャートのステップＳ２１７における定義式による分割データの生成処理、具体的には分割数n=3の場合の分割データの生成処理について詳しく説明する。

【００５１】

まず、整数倍を意味する変数m=0の場合には、ステップＳ２１７に示す定義式から各分割データD(i)=D(1)～D(3)の各々を構成する各分割部分データD(i,2×m+j)=D(i,j)(i=1～3,j=1～2)は、次のようになる。

【００５２】

D(1,1)=S(1)*Q(1,1,1)*Q(1,1,2)

D(1,2)=S(2)*Q(2,1,1)*Q(2,1,2)

D(2,1)=S(1)*Q(1,2,1)*Q(1,2,2)

D(2,2)=S(2)*Q(2,2,1)*Q(2,2,2)

D(3,1)=R(1)

D(3,2)=R(2)

上記の６つの式のうち上から４つの式に含まれるQ(j,i,k)を具体的に求める。これはc(j,i,k)を2×2行列であるU[2,2]×(P[2,2])^(j-1)のi行k列の値としたとき下記のように定義される。

【００５３】

c(j,i,k)=1 のとき Q(j,i,k)=R(k)

c(j,i,k)=0 のとき Q(j,i,k)=0

ここで、

j=1のときは

【数８】

$$U[2,2]\times(P[2,2])^{(j-1)} = U[2,2]\times(P[2,2])^{0}$$
$$= U[2,2]\times E[2,2]$$
$$= U[2,2]$$
$$= \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}$$

j=2のときは

【数９】

$$U[2,2]\times(P[2,2])^{(j-1)} = U[2,2]\times(P[2,2])^{1}$$
$$= U[2,2]\times P[2,2]$$
$$= \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix} \times \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$
$$= \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}$$

これを用いると、各分割部分データD(i,j)は次のような定義式により生成される。

【００５４】

【数１０】

D(1,1)=S(1)*Q(1,1,1)*Q(1,1,2)=S(1)*R(1)*R(2)

D(1,2)=S(2)*Q(2,1,1)*Q(2,1,2)=S(2)*R(1)*R(2)

D(2,1)=S(1)*Q(1,2,1)*Q(1,2,2)=S(1)*R(1)*0=S(1)*R(1)

D(2,2)=S(2)*Q(2,2,1)*Q(2,2,2)=S(2)*0*R(2)=S(2)*R(2)

上述した各分割部分データD(i,j)を生成するための定義式は、図３にも図示されている。

【００５５】

図３は、上述したように１６ビットの元データSを８ビットの処理単位ビット

長に基づいて分割数n=3で３分割する場合の各データと定義式および各分割部分データから元データを復元する場合の計算式などを示す表である。

【００５６】

ここで、上述した定義式により分割データD(1),D(2),D(3)および各分割部分データD(1,1),D(1,2),D(2,1),D(2,2),D(3,1),D(3,2)を生成する過程と定義式の一般形について説明する。

【００５７】

まず、第１の分割データD(1)に対しては、第１の分割部分データD(1,1)は、上述した定義式S(1)*R(1)*R(2)で定義され、第２の分割部分データD(1,2)は定義式S(2)*R(1)*R(2)で定義される。なお、この定義式の一般形は、D(1,j)に対してはS(j)*R(j)*R(j+1)であり、D(1,j+1)に対してS(j+1)*R(j)*R(j+1)である（jは奇数とする）。定義式に従って計算すると、D(1,1)は00110110, D(1,2)は10110011となるので、D(1)は00110110 10110011である。なお、定義式の一般形は、図４にまとめて示されている。

【００５８】

また、第２の分割データD(2)に対しては、D(2,1)はS(1)*R(1)で定義され、D(2,2)はS(2)*R(2)で定義される。この定義式の一般形は、D(2,j)に対してはS(j)*R(j)であり、D(2,j+1)に対してはS(j+1)*R(j+1)である（jは奇数とする）。定義式に従って計算すると、D(2,1)は00000011, D(2,2)は00000010となるので、D(2)は00000011 00000010である。

【００５９】

更に第３の分割データD(3)に対しては、D(3,1)はR(1)で定義され、D(3,2)はR(2)で定義される。この定義式の一般形は、D(3,j)に対してはR(j)であり、D(3,j+1)に対してはR(3,j+1)である（jは奇数とする）。定義式に従って計算すると、D(3,1)は10110001, D(3,2)は00110101となるので、D(3)は10110001 00110101である。

【００６０】

上記説明は、S,R,D(1),D(2),D(3)の長さを１６ビットとしたが、データの先頭から上記分割処理を繰り返すことにより、どのような長さの元データSからでも

分割データD(1),D(2),D(3)を生成することができる。また、処理単位ビット長bは任意にとることができ、元データSの先頭から順にb×2の長さ毎に上記分割処理を繰り返すことにより任意の長さの元データ、具体的には処理単位ビット長b×2の整数倍の長さの元データに対して適用することができる。なお、元データSの長さが処理単位ビット長b×2の整数倍でない場合は、例えば、データ末尾の部分を０で埋めるなどして元データSの長さを処理単位ビット長b×2の整数倍に合わせることにより上述した本実施形態の分割処理を適用することができる。

【００６１】

次に、図３の右側に示す表を参照して、分割データから元データを復元する処理について説明する。

【００６２】

まず、利用者は端末５からネットワーク３を介して分割装置１にアクセスし、分割装置１のデータ送受信手段１７を介して元データSの復元を要求する。分割装置１は、この元データSの復元要求を受け取ると、この元データSに対応する分割データD(1),D(2),D(3)が保管サーバ７ａ，７ｂ，７ｃに保管されていることを記憶しているので、ネットワーク３を介して保管サーバ７ａ，７ｂ，７ｃから分割データD(1),D(2),D(3)を取得し、この取得した分割データD(1),D(2),D(3)から次に示すように元データSを復元する。

【００６３】

まず、分割部分データD(2,1),D(3,1)から第１の元部分データS(1)を次のように生成することができる。

【００６４】

$$D(2,1)*D(3,1)=(S(1)*R(1))*R(1)$$
$$=S(1)*(R(1)*R(1))$$
$$=S(1)*0$$
$$=S(1)$$

具体的に計算すると、D(2,1)は00000011，D(3,1)は10110001なので、S(1)は10110010となる。

【００６５】

また、別の分割部分データから次のように第２の元部分データS(2)を生成することができる。

【００６６】

D(2,2)*D(3,2)=(S(2)*R(2))*R(2)

=S(2)*(R(2)*R(2))

=S(2)*0

=S(2)

具体的に計算すると、D(2,2)は00000010，D(3,2)は00110101なので、S(2)は00110111となる。

【００６７】

一般に、jを奇数として、

D(2,j)*D(3,j)=(S(j)*R(j))*R(j)

=S(j)*(R(j)*R(j))

=S(j)*0

=S(j)

であるから、D(2,j)*D(3,j)を計算すれば、S(j)が求まる。

【００６８】

また、一般に、jを奇数として、

【数１１】

D(2,j+1)*D(3,j+1)=(S(j+1)*R(j+1))*R(j+1)

=S(j+1)*(R(j+1)*R(j+1))

=S(j+1)*0

=S(j+1)

であるから、D(2,j+1)*D(3,j+1)を計算すれば、S(j+1)が求まる。

【００６９】

次に、D(1),D(3)を取得してSを復元する場合には、次のようになる。

【００７０】

【数１２】

D(1,1)*D(3,1)*D(3,2)=(S(1)*R(1)*R(2))*R(1)*R(2)

=S(1)*(R(1)*R(1))*(R(2)*R(2))

=S(1)*0*0

=S(1)

であるから、D(1,1)*D(3,1)*D(3,2)を計算すれば、S(1)が求まる。具体的に計算すると、D(1,1)は00110110，D(3,1)は10110001，D(3,2)は00110101なので、S(1)は10110010となる。

【００７１】

また同様に、

【数１３】

$$D(1,2)*D(3,1)*D(3,2)=(S(2)*R(1)*R(2))*R(1)*R(2)$$
$$=S(2)*(R(1)*R(1))*(R(2)*R(2))$$
$$=S(2)*0*0$$
$$=S(2)$$

であるから、D(1,2)*D(3,1)*D(3,2)を計算すれば、S(2)が求まる。具体的に計算すると、D(1,2)は10110011，D(3,1)は10110001，D(3,2)は00110101なので、S(2)は00110111となる。

【００７２】

一般に、jを奇数として、

【数１４】

$$D(1,j)*D(3,j)*D(3,j+1)=(S(j)*R(j)*R(j+1))*R(j)*R(j+1)$$
$$=S(j)*(R(j)*R(j))*(R(j+1)*R(j+1))$$
$$=S(j)*0*0$$
$$=S(j)$$

であるから、D(1,j)*D(3,j)*D(3,j+1)を計算すれば、S(j)が求まる。

【００７３】

また、一般に、jを奇数として、

【数１５】

$$D(1,j+1)*D(3,j)*D(3,j+1)=(S(j+1)*R(j)*R(j+1))*R(j)*R(j+1)$$
$$=S(j+1)*(R(j)*R(j))*(R(j+1)*R(j+1))$$

=S(j+1)*0*0

=S(j+1)

であるから、D(1,j+1)*D(3,j)*D(3,j+1)を計算すれば、S(j+1)が求まる。

【００７４】

次に、D(1),D(2)を取得してSを復元する場合には、次のようになる。

【００７５】

【数１６】

$$D(1,1)*D(2,1)=(S(1)*R(1)*R(2))*(S(1)*R(1))$$
$$=(S(1)*S(1))*(R(1)*R(1))*R(2)$$
$$=0*0*R(2)$$
$$=R(2)$$

であるから、D(1,1)*D(2,1)を計算すれば、R(2)が求まる。具体的に計算すると、D(1,1)は00110110, D(2,1)は00000011なので、R(2)は00110101となる。

【００７６】

また同様に、

【数１７】

$$D(1,2)*D(2,2)=(S(2)*R(1)*R(2))*(S(2)*R(2))$$
$$=(S(2)*S(2))*R(1)*(R(2)*R(2))$$
$$=0*R(1)*0$$
$$=R(1)$$

であるから、D(1,2)*D(2,2)を計算すれば、R(1)が求まる。具体的に計算すると、D(1,2)は10110011, D(2,2)は00000010なので、R(1)は10110001となる。

【００７７】

このR(1),R(2)を使用してS(1),S(2)を求める。

【００７８】

$$D(2,1)*R(1)=(S(1)*R(1))*R(1)$$
$$=S(1)*(R(1)*R(1))$$
$$=S(1)*0$$
$$=S(1)$$

であるから、D(2,1)*R(1)を計算すれば、S(1)が求まる。具体的に計算すると、D(2,1)は00000011，R(1)は10110001なので、S(1)は10110010となる。

【００７９】

また同様に、

$$D(2,2)*R(2)=(S(2)*R(2))*R(2)$$
$$=S(2)*(R(2)*R(2))$$
$$=S(2)*0$$
$$=S(2)$$

であるからD(2,2)*R(2)を計算すればS(2)が求まる。具体的に計算するとD(2,2)は00000010，R(2)は00110101なので、S(2)は00110111となる。

【００８０】

一般に、jを奇数として、

【数１８】

$$D(1,j)*D(2,j)=(S(j)*R(j)*R(j+1))*(S(j)*R(j))$$
$$=(S(j)*S(j))*(R(j)*R(j))*R(j+1)$$
$$=0*0*R(j+1)$$
$$=R(j+1)$$

であるからD(1,j)*D(2,j)を計算すればR(j+1)が求まる。

【００８１】

また同様に、

【数１９】

$$D(1,j+1)*D(2,j+1)=(S(j+1)*R(j)*R(j+1))*(S(j+1)*R(j+1))$$
$$=(S(j+1)*S(j+1))*R(j)*(R(j+1)*R(j+1))$$
$$=0*R(j)*0$$
$$=R(j)$$

であるからD(1,j+1)*D(2,j+1)を計算すればR(j)が求まる。

【００８２】

このR(j),R(j+1)を使用してS(j),S(j+1)を求める。

【００８３】

D(2,j)*R(j)=(S(j)*R(j))*R(j)

=S(j)*(R(j)*R(j))

=S(j)*0

=S(j)

であるからD(2,j)*R(j)を計算すればS(j)が求まる。

【００８４】

また同様に、

D(2,j+1)*R(j+1)=(S(j+1)*R(j+1))*R(j+1)

=S(j+1)*(R(j+1)*R(j+1))

=S(j+1)*0

=S(j+1)

であるからD(2,j+1)*R(j+1)を計算すればS(j+1)が求まる。

【００８５】

上述したように、元データの先頭から処理単位ビット長bに基づいて分割処理を繰り返し行って、分割データを生成した場合には、３つの分割データD(1),D(2),D(3)のすべてを用いなくても、３つの分割データのうち、２つの分割データを用いて上述したように元データを復元することができる。

【００８６】

本発明の他の実施形態として、乱数Rのビット長を元データSのビット長よりも短いものを使用して、元データの分割処理を行うことができる。

【００８７】

すなわち、上述した乱数RはS,D(1),D(2),D(3)と同じビット長のデータとしたが、乱数Rを元データSのビット長より短いものとし、分割データD(1),D(2),D(3)の生成にこの短いビット長の乱数Rを繰り返し用いるものである。

【００８８】

なお、分割データD(3)は乱数Rのみから生成されるので、分割データD(3)は乱数Rを繰り返して保管しておく必要はない。例えば、元データSのビット長を１６００ビット（200バイト）としたとき、乱数Rは任意にとった１６０ビット（20バイト）のデータの繰り返しとする。つまり、R(1)～R(20)はランダムに生成し、R

(21)～R(200)はR(21)=R(1),R(22)=R(2),…,R(40)=R(20),R(41)=R(1),R(42)=R(2),…,R(60)=R(20),R(61)=R(1),R(62)=R(2),…,R(80)=R(20),……,R(181)=R(1),R(182)=R(2),…,R(200)=R(20)とする。

【００８９】

先の説明では、分割部分データD(3,j)を乱数部分データR(j)と定義してD(3)を生成しているが、D(3,20)まで保管すれば十分である。つまり、D(3)の長さはD(1),D(2)の１０分の１となる。従って、保管すべきデータの総量は先の実施形態では元データSの３倍であるが、この実施形態では２．１倍とすることができる。乱数Rにおける繰り返し部分のデータの長さは、短すぎると、D(1)またはD(2)のみからRが解読されてしまうことも考えられるため、適切な長さを選択することが望ましい。

【００９０】

上述した各実施形態は、元データを３つに分割し、そのうち２つから元データが復元可能となるものであったが、分割数nを３より大きくとって、n個より少ない個数の分割データから元データを復元することができることは勿論のことである。

【００９１】

その１つの応用例として、元データを４つの分割データに分割する分割数n=4の場合の分割処理について図５に示すフローチャートおよび図６に示す定義式の一般形などを参照して説明する。

【００９２】

まず、利用者は端末５から分割装置１にアクセスして元データSを送信し、分割装置１ではデータ送受信手段１７が端末５からの元データSを受信し、分割装置１に供給する（ステップＳ３０１）。それから、利用者は端末５から分割数nとして４を分割装置１に指示する（ステップＳ３０３）。この分割数nは分割装置１において予め定められた値を用いてもよい。また、処理単位ビット長bが一例として８ビットと決定される（ステップＳ３０５）。次に、元データSのビット長が8×3の整数倍であるか否かを判定し、整数倍でない場合には、元データSの末尾を０で埋める（ステップＳ３０７）。また、整数倍を意味する変数mを０

に設定する（ステップＳ３０９）。

【００９３】

次に、元データSの8×3×m+1ビット目から8×3ビット分のデータが存在するか否かが判定される（ステップＳ３１１）。なお、本例では、元データSが8×3=24ビット長のデータの場合について説明している。

【００９４】

ステップＳ３１１の判定の結果、本例の元データSでは8×3=24ビットのデータであり、変数m=1の場合に相当する8×3×m(=1)+1ビットに相当する２５ビット以降のデータは存在しないので、ステップＳ３１１からステップＳ３２１に進むことになるが、今の場合は、変数mは０であるので、元データSの8×3×m+1ビット目は、8×3×0+1=1となり、元データSの２４ビットの１ビット目から8×3ビット分にデータが存在するため、ステップＳ３１３に進む。

【００９５】

ステップＳ３１３では、変数jを１から３（=分割数n-1）まで変えて、元データSの8×(3×m+j-1)+1ビット目から８ビット分（=処理単位ビット長）のデータを元部分データS(3×m+j)に設定し、これにより元データSを処理単位ビット長で区分けした３個の元部分データS(1),S(2),S(3)が生成される。

【００９６】

次に、変数jを１から３まで変えて、乱数部分データR(3×m+j)に乱数発生手段１５から発生する８ビットの長さの乱数を設定し、これにより乱数Rを処理単位ビット長で区分けした３個の乱数部分データR(1),R(2),R(3)が生成される（ステップＳ３１５）。

【００９７】

次に、ステップＳ３１７において、乱数iを１から４(=分割数n)まで変えるとともに、更に各変数iにおいて変数jを１から３(=分割数n-1）まで変えながら、ステップＳ３１７に示す分割データを生成するための定義式により複数の分割データD(i)の各々を構成する各分割部分データD(i,3×m+j)を生成する。この結果、次に示すような分割データDが生成される。

【００９８】

【数２０】

分割データD

=4個の分割データD(i)=D(1),D(2),D(3),D(4)

第１の分割データD(1)

=3個の分割部分データD(1,j)=D(1,1),D(1,2),D(1,3)

第２の分割データD(2)

=3個の分割部分データD(2,j)=D(2,1),D(2,2),D(2,3)

第３の分割データD(3)

=3個の分割部分データD(3,j)=D(3,1),D(3,2),D(3,3)

第４の分割データD(4)

=3個の分割部分データD(4,j)=D(4,1),D(4,2),D(4,3)

なお、各分割部分データD(i,j)を生成するためのステップＳ３１７に示す定義式は、本例のように分割数n=4の場合には、具体的には図６に示す表に記載されているものとなる。図６に示す表から、分割部分データD(1,1)を生成するための定義式はS(1)*R(1)*R(2)*R(3)であり、D(1,2)の定義式はS(2)*R(1)*R(2)*R(3)であり、D(1,3)の定義式はS(3)*R(1)*R(2)*R(3)であり、D(2,1)の定義式はS(1)*R(1)*R(2)であり、D(2,2)の定義式はS(2)*R(2)*R(3)であり、D(2,3)の定義式はS(3)*R(1)*R(3)であり、D(3,1)の定義式はS(1)*R(1)であり、D(3,2)の定義式はS(2)*R(2)であり、D(3,3)の定義式はS(3)*R(3)であり、D(4,1)の定義式はR(1)であり、D(4,2)の定義式はR(2)であり、D(4,3)の定義式はR(3)である。また、図６に示す表にはm>0の場合の任意の整数についての一般的な定義式も記載されている。

【００９９】

このように変数m=0の場合について分割データDを生成した後、次に変数mを１増やし（ステップＳ３１９）、ステップＳ３１１に戻り、変数m=1に該当する元データSの２５ビット以降について同様の分割処理を行おうとするが、本例の元データSは２４ビットであり、２５ビット以降のデータは存在しないので、ステップＳ３１１からステップＳ３２１に進み、上述したように生成した分割データD(1),D(2),D(3),D(4)を分割装置１のデータ送受信手段１７からネットワーク３を介して保管サーバ７にそれぞれ送信し、各保管サーバ７に保管し、分割処理を

終了する。図１では保管サーバは３個であるが、分割数に応じて保管サーバを増やし、各分割データを異なる保管サーバに保管することが望ましい。

【０１００】

ここで、上述した図５のフローチャートのステップＳ３１７における定義式による分割データの生成処理、具体的には分割数n=4の場合の分割データの生成処理について詳しく説明する。

【０１０１】

まず、ステップＳ３１７に示す定義式から各分割データD(i)=D(1)～D(4)の各々を構成する各分割部分データD(i,3×m+j)は、次のようになる。

【０１０２】

【数２１】

$$D(1,3\times m+1)=S(3\times m+1)*Q(1,1,1)*Q(1,1,2)*Q(1,1,3)$$
$$D(1,3\times m+2)=S(3\times m+2)*Q(2,1,1)*Q(2,1,2)*Q(2,1,3)$$
$$D(1,3\times m+3)=S(3\times m+3)*Q(3,1,1)*Q(3,1,2)*Q(3,1,3)$$
$$D(2,3\times m+1)=S(3\times m+1)*Q(1,2,1)*Q(1,2,2)*Q(1,2,3)$$
$$D(2,3\times m+2)=S(3\times m+2)*Q(2,2,1)*Q(2,2,2)*Q(2,2,3)$$
$$D(2,3\times m+3)=S(3\times m+3)*Q(3,2,1)*Q(3,2,2)*Q(3,2,3)$$
$$D(3,3\times m+1)=S(3\times m+1)*Q(1,3,1)*Q(1,3,2)*Q(1,3,3)$$
$$D(3,3\times m+2)=S(3\times m+2)*Q(2,3,1)*Q(2,3,2)*Q(2,3,3)$$
$$D(3,3\times m+3)=S(3\times m+3)*Q(3,3,1)*Q(3,3,2)*Q(3,3,3)$$
$$D(4,3\times m+1)=R(3\times m+1)$$
$$D(4,3\times m+2)=R(3\times m+2)$$
$$D(4,3\times m+3)=R(3\times m+3)$$

次に、Q(j,i,k)を具体的に求める。これはc(j,i,k)を3×3行列であるU[3,3]×(P[3,3])^(j-1)のi行k列の値としたとき下記のように定義される。

【０１０３】

c(j,i,k)=1 のとき Q(j,i,k)=R(3×m+k)

c(j,i,k)=0 のとき Q(j,i,k)=0

j=1のときは

【数２２】

$$U[3,3]\times(P[3,3])^{(j-1)} = U[3,3]\times(P[3,3])^{0} = U[3,3]$$

$$= \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 0 \end{pmatrix}$$

j=2のときは

【数２３】

$$U[3,3]\times(P[3,3])^{(j-1)} = U[3,3]\times(P[3,3])^{1} = U[3,3]\times(P[3,3])$$

$$= \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 0 \end{pmatrix} \times \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 1 & 0 \end{pmatrix}$$

j=3のときは

【数２４】

$$U[3,3]\times(P[3,3])^{(j-1)} = U[3,3]\times(P[3,3])^{2} = U[3,3]\times(P[3,3])\times(P[3,3])$$

$$= \begin{pmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 1 & 0 \end{pmatrix} \times \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 0 & 1 \\ 0 & 0 & 1 \end{pmatrix}$$

これを用いると、各分割部分データは次のような定義式により生成される。

【０１０４】

【数２５】

$$D(1,3\times m+1)=S(3\times m+1)*Q(1,1,1)*Q(1,1,2)*Q(1,1,3)$$
$$=S(3\times m+1)*R(3\times m+1)*R(3\times m+2)*R(3\times m+3)$$
$$D(1,3\times m+2)=S(3\times m+2)*Q(2,1,1)*Q(2,1,2)*Q(2,1,3)$$
$$=S(3\times m+2)*R(3\times m+1)*R(3\times m+2)*R(3\times m+3)$$
$$D(1,3\times m+3)=S(3\times m+3)*Q(3,1,1)*Q(3,1,2)*Q(3,1,3)$$
$$=S(3\times m+3)*R(3\times m+1)*R(3\times m+2)*R(3\times m+3)$$

D(2,3×m+1)=S(3×m+1)*Q(1,2,1)*Q(1,2,2)*Q(1,2,3)
=S(3×m+1)*R(3×m+1)*R(3×m+2)
D(2,3×m+2)=S(3×m+2)*Q(2,2,1)*Q(2,2,2)*Q(2,2,3)
=S(3×m+2)*R(3×m+2)*R(3×m+3)
D(2,3×m+3)=S(3×m+3)*Q(3,2,1)*Q(3,2,2)*Q(3,2,3)
=S(3×m+3)*R(3×m+1)*R(3×m+3)
D(3,3×m+1)=S(3×m+1)*Q(1,3,1)*Q(1,3,2)*Q(1,3,3)
=S(3×m+1)*R(3×m+1)
D(3,3×m+2)=S(3×m+2)*Q(2,3,1)*Q(2,3,2)*Q(2,3,3)
=S(3×m+2)*R(3×m+2)
D(3,3×m+3)=S(3×m+3)*Q(3,3,1)*Q(3,3,2)*Q(3,3,3)
=S(3×m+3)*R(3×m+3)
D(4,3×m+1)=R(3×m+1)
D(4,3×m+2)=R(3×m+2)
D(4,3×m+3)=R(3×m+3)

ここで、上述したように図２のステップＳ２１７や図５のステップＳ３１７で示した定義式に基づいて元データを分割する分割規則について一般的な表現で記載する。

【０１０５】

まず、元データ、乱数、分割データ、分割数および処理単位ビット長をそれぞれS,R,D,nおよびbで表すとともに、複数n個のうちの１つを表わす変数としてi(=1～n)およびj(=1～n-1)を用いて複数(n-1)個の元部分データ、複数(n-1)個の乱数部分データ、複数(n)個の分割データおよび各分割データの複数(n-1)個の分割部分データのそれぞれのうちの１つをそれぞれS(j),R(j),D(i)およびD(i,j)で表わす。

【０１０６】

それから、上記変数jを１からn-1まで変えて、各元部分データS(j)を元データSのb×(j-1)+1ビット目からbビット分のデータとして作成する。次に、U[n,n]をn×n行列である上三角行列とし、P[n,n]をn×n行列である回転行列としたとき、

c(j,i,k)を(n-1)×(n-1)行列であるU[n-1,n-1]×(P[n-1,n-1]＾(j-1)のi行k列の値と定義する。そして、c(j,i,k)=1のとき、Q(j,i,k)=R(k)，c(j,i,k)=0のとき、Q(j,i,k)=0と定義したとき、変数iを１からnまで変えながら、各変数iにおいて変数jを１からn-1まで変えた場合において、

i<nのとき、各分割部分データD(i,j)を

【数２６】

$$D(i,j) = S(j)*\left(\prod_{k=1}^{n-1} Q(j,i,k)\right)$$

と設定し、またi=nのとき、各分割部分データD(i,j)を

D(i,j)=R(j)

と設定する。上記処理を元データSの先頭から末尾まで繰り返し行うことにより元データSから分割数nの分割データを生成することができる。

【０１０７】

次に、上述したように元データSを４分割して生成された分割データD(1),D(2),D(3),D(4)から元データSを復元する処理について図６を参照して説明する。なお、図６に示す４分割の場合には、変数jを3×m+1(m≧0である任意の整数）として、同図に示す一般的な定義式から次に示すように元データSを生成することができる。

【０１０８】

まず、分割データD(1),D(2)から元データSを求める場合について説明する。

【０１０９】

【数２７】

$$\begin{aligned} D(1,j)*D(2,j) &= (S(j)*R(j)*R(j+1)*R(j+2))*((S(j)*R(j)*R(j+1)) \\ &= (S(j)*S(j))*(R(j)*R(j))*(R(j+1)*R(j+1))*R(j+2) \\ &= 0*0*0*R(j+2) \\ &= R(j+2) \end{aligned}$$

従って、D(1,j)*D(2,j)を計算すれば、乱数R(j+2)が求まり、同様にD(1,j+1)*D(2,j+1)を計算すれば、乱数R(j)が求まり、同様にD(1,j+2)*D(2,j+2)を計算すれば、乱数R(j+1)が求まり、これらの得られた乱数R(j),R(j+1),R(j+2)を用いれ

ば、

【数２８】

$$D(1,j)*R(j)*R(j+1)*R(j+2)$$
$$=(S(j)*R(j)*R(j+1)*R(j+2))*(R(j)*R(j+1)*R(j+2))$$
$$=S(j)*(R(j)*R(j+1))*(R(j+1)*R(j+1))*(R(j+2)*R(j+2))$$
$$=S(j)*0*0*0$$
$$=S(j)$$

であるから、D(1,j)*R(j)*R(j+1)*R(j+2)を計算して、S(j)を求めてもよいし、D(2,j)*R(j)*R(j+1)からS(j)を求めることもできる。

【０１１０】

同様に、D(1,j+1)*R(j)*R(j+1)*R(j+2)またはD(2,j+1)*R(j+1)*R(j+2)を計算してS(j+1)を求めることができ、また同様にD(1,j+2)*R(j)*R(j+1)*R(j+2)またはD(2,j+2)*R(j)*R(j+2)を計算してS(j+2)を求めることができる。

【０１１１】

更に、上述したと同様に、D(2)とD(3)からSを求めることができる。

【０１１２】

具体的には、まずR(j),R(j+1),R(j+2)を求めてから、D(2,j),D(2,j+1),D(2,j+2)またはD(3,j),D(3,j+1),D(3,j+2)とR(j),R(j+1),R(j+2)のXOR演算によりS(j),S(j+1),S(j+2)を求めることができる。

【０１１３】

また更に、D(1)とD(4)またはD(2)とD(4)またはD(3)とD(4)からSを求めることができる。

【０１１４】

D(4)はRをそのものから定義したものであるから、計算することなくD(4)からR(j),R(j+1),R(j+2)を取得することができ、例えば、D(1,j),D(1,j+1),D(1,j+2)とR(j),R(j+1),R(j+2)のXOR演算によりS(j),S(j+1),S(j+2)を求めることができる。

【０１１５】

上述したように、演算回数の差が１である任意の２つの分割データD(1)とD(2)

、または、D(2)とD(3)、または、D(4)と任意の１つの分割データD(1)またはD(2)またはD(3)からSが復元可能である。すなわち、４つの分割データの中から任意に３つの分割データを取得すれば、その中には必ず上述したいずれかのケースが含まれるため、４つのうち任意の３つの分割データから元データを復元可能である。

【０１１６】

図７は、５分割の場合の分割データと定義式を示す表である。この５分割の場合は、jを4×m+1(mはm≧0である任意の整数) として、分割データの定義式から、上述した４分割の場合の復元処理と同様のことが言える。従って、演算回数の差が１である任意の２つの分割データD(1)とD(2)、または、D(2)とD(3)、または、D(3)とD(4)、または、D(5)と任意の１つの分割データD(1)またはD(2)またはD(3)またはD(4)から元データSが復元可能である。そして、５つの分割データの中から任意に３つの分割データを取得すれば、その中には必ずこのいずれかのケースが含まれるため、５つのうち任意の３つから復元可能であるといえる。

【０１１７】

また、分割数nを５より大きくとった場合も同様にして分割データを構成すれば、nが奇数である場合は(n+1)/2個、nが偶数である場合は(n/2)+1個の分割データから元データを復元することができる。この個数は、n個の分割データがあったときに、隣り合ったものを選択せず、かつ、n個目の分割データを選択しないような最大個数に１を加えたものである。つまり、前記最大個数に１を加えれば演算回数の差が１である２つの分割データまたはn個目の分割データとその他のデータを必ず含むこととなるため、復元に必要な個数が前記のとおりといえる。

【０１１８】

次に、図８に示すフローチャートを参照して、分割数がnで、処理単位ビット長がbである場合の一般的な分割処理について説明する。

【０１１９】

まず、利用者は端末５から分割装置１にアクセスして元データSを送信し、分割装置１ではデータ送受信手段１７が端末５からの元データSを受信し、分割装置１に供給する（ステップＳ４０１）。また、利用者は端末５から分割数n(n≧3

である任意の整数）を分割装置１に指示する（ステップＳ４０３）。この分割数nは分割装置１において予め定められた値を用いてもよい。処理単位ビット長bを決定する（ステップＳ４０５）。なお、bは０より大きい任意の整数である。次に、元データSのビット長がb×(n-1)の整数倍であるか否かを判定し、整数倍でない場合には、元データSの末尾を０で埋める（ステップＳ４０７）。また、整数倍を意味する変数mを０に設定する（ステップＳ４０９）。

【０１２０】

次に、元データSのb×(n-1)×m+1ビット目からb×(n-1)ビット分のデータが存在するか否かが判定される（ステップＳ４１１）。この判定の結果、データが存在しない場合は、ステップＳ４２１に進むことになるが、今の場合は、ステップＳ４０９で変数mは０に設定された場合であるので、データが存在するため、ステップＳ４１３に進む。

【０１２１】

ステップＳ４１３では、変数jを１からn-1まで変えて、元データSのb×(n-1)×m+j-1)+1ビット目からbビット分のデータを元部分データS((n-1)×m+j)に設定する処理を繰り返し、これにより元データSを処理単位ビット長bで区分けした(n-1)個の元部分データS(1),S(2),…S(n-1)が生成される。

【０１２２】

次に、変数jを１からn-1まで変えて、乱数部分データR((n-1)×m+j)に乱数発生手段１５から発生する処理単位ビット長bの乱数を設定し、これにより乱数Rを処理単位ビット長bで区分けしたn-1個の乱数部分データR(1),R(2),…R(n-1)が生成される（ステップＳ４１５）。

【０１２３】

次に、ステップＳ４１７において、変数iを１からnまで変えるとともに、更に各変数iにおいて変数jを１からn-1まで変えながら、ステップＳ４１７に示す分割データを生成するための定義式により複数の分割データD(i)の各々を構成する各分割部分データD(i,(n-1)×m+j)を生成する。この結果、次に示すような分割データDが生成される。

【０１２４】

【数２９】

分割データD

=n個の分割データD(i)=D(1),D(2),…D(n)

第１の分割データD(1)

=n-1個の分割部分データD(1,j)=D(1,1),D(1,2),…D(1,n-1)

第２の分割データD(2)

=n-1個の分割部分データD(2,j)=D(2,1),D(2,2),…D(2,n-1)

… … …

… … …

第nの分割データD(n)

=n-1個の分割部分データD(n,j)=D(n,1),D(n,2),…D(n,n-1)

このように変数m=0の場合について分割データDを生成した後、次に変数mを１増やし（ステップＳ４１９）、ステップＳ４１１に戻り、変数m=1に該当する元データSのb×(n-1)ビット以降について同様の分割処理を行う。最後にステップＳ４１１の判定の結果、元データSにデータがなくなった場合、ステップＳ４１１からステップＳ４２１に進み、上述したように生成した分割データDを分割装置１のデータ送受信手段１７からネットワーク３を介して保管サーバ７にそれぞれ送信し、各保管サーバ７に保管し、分割処理を終了する。図１では保管サーバは３個であるが、分割数に応じて保管サーバを増やし、各分割データを異なる保管サーバに保管することが望ましい。

【０１２５】

次に、図９に示すフローチャートを参照して、分割数nが２の場合の分割処理について説明する。すなわち、上述した各実施形態は図８のフローチャートのステップＳ４０３に示したように分割数nが３以上(n≧3)の場合についてのものであるので、図９を用いて分割数nが２の場合について説明する。

【０１２６】

まず、利用者は端末５から分割装置１にアクセスして元データSを分割装置１に供給する（ステップＳ５０１）。また、利用者は端末５から分割数nとして２を分割装置１に指示する（ステップＳ５０３）。この分割数nは分割装置１にお

いて予め定められた値を用いてもよい。それから処理単位ビット長bとして８ビットを決定する（ステップＳ５０５）。次に、元データSのビット長が８の整数倍であるか否かを判定し、整数倍でない場合には、元データSの末尾を０で埋める（ステップＳ５０７）。また、整数倍を意味する変数mを０に設定する（ステップＳ５０９）。

【０１２７】

次に、元データSの8×m+1ビット目から８ビット分のデータが存在するか否かが判定される（ステップＳ５１１）。この判定の結果、データが存在しない場合は、ステップＳ５２１に進むことになるが、今の場合は、変数mは０に設定されているので、データが存在するため、ステップＳ５１３に進む。

【０１２８】

ステップＳ５１３では、元データSの8×m+1ビット目から８ビット分のデータを元部分データS(m+1)に設定し、これにより元部分データS(1)が生成される。

【０１２９】

次に、乱数部分データR(m+1)に乱数発生手段１５から発生する８ビットの乱数を設定し、これにより乱数部分データR(1)が生成される（ステップＳ５１５）。

【０１３０】

次に、ステップＳ５１７において、同ステップに示す定義式により分割データDの各々を構成する各分割データD(1,m+1),D(2,m+1)が生成される。

【０１３１】

このように変数m=0の場合について分割データDを生成した後、次に変数mを１増やし（ステップＳ５１９）、ステップＳ５１１に戻り、変数m=1に該当する元データSの８ビット以降について同様の分割処理を行う。最後にステップＳ５１１の判定の結果、元データSにデータがなくなった場合、ステップＳ５１１からステップＳ５２１に進み、上述したように生成した分割データD(1)からD(2)を分割装置１のデータ送受信手段１７からネットワーク３を介して保管サーバ７にそれぞれ送信し、各保管サーバ７に保管し、分割処理を終了する。図１では保管サーバは３個であるが、このうち２個の保管サーバに各分割データを保管すればよい。

【０１３２】

ここにおいて、上述した図９のフローチャートのステップＳ５１７における定義式による分割データの生成処理、具体的には分割数n=2の場合の分割データの生成処理について詳しく説明する。

【０１３３】

変数m=0の場合には、ステップＳ５１７に示す定義式から各分割データD(1,1), D(2,1)は、次のようになる。

【０１３４】

D(1,1)=S(1)*Q(1,1,1)

D(2,1)=R(1)

次に、Q(j,i,k)を具体的に求める。ここで、n=2を定義に当てはめると、j,i,kはいずれも１しか値をとらない。

【０１３５】

c(j,i,k)は1×1行列であるU[1,1]×(P[1,1])＾(j-1)のi行k列の値としたとき下記のように定義される。

【０１３６】

【数３０】

$$c(j,i,k)=1 \text{ のとき } Q(j,i,k)=R(k)$$

$$c(j,i,k)=0 \text{ のとき } Q(j,i,k)=0$$

$$U[1,1]\times(P[1,1])^{(j-1)}=U[1,1]\times(P[1,1])^{0}$$

$$=(1)\times E[1,1]$$

$$=(1)\times(1)$$

$$=(1)$$

従って、c(1,1,1)は１であるから、Q(1,1,1)はR(1)と定義される。

【０１３７】

以上から定義式は

D(1,1)=S(1)*R(1)

D(2,1)=R(1)

となる。変数mを使用した形式では、

D(1,m+1)=S(m+1)*R(m+1)

D(2,m+1)=R(m+1)

となる。

【０１３８】

なお、分割数n=2の場合には、２個の分割データのうち、どちらか一方を取得しただけでは、元データSを復元することはできず、２個のすべての分割データを取得して元データSを復元することになる。

【０１３９】

なお、上記実施形態のデータ分割方法の処理手順をプログラムとして例えばＣＤやＦＤなどの記録媒体に記録して、この記録媒体をコンピュータシステムに組み込んだり、または記録媒体に記録されたプログラムを通信回線を介してコンピュータシステムにダウンロードしたり、または記録媒体からインストールし、該プログラムでコンピュータシステムを作動させることにより、データ分割方法を実施するデータ分割装置として機能させることができることは勿論であり、このような記録媒体を用いることにより、その流通性を高めることができるものである。

【０１４０】

上述してきたように、本実施例によれば、所定の定義式が元部分データと乱数部分データの排他的論理和からなるので、従来のように多項式や剰余演算を行う高速かつ高性能な演算処理能力を必要とせず、大容量のデータに対しても簡単な演算処理を繰り返して分割データを簡単かつ迅速に生成することができる。

【０１４１】

また、生成した複数の分割データのうち分割数よりも少ない数の分割データに対して定義式を適用することにより元データを復元するので、分割数よりも少ない任意の数xの分割データで元データを復元でき、分割数からxを減算した数までの分割データを紛失したり破壊したとしても、元データを復元することができる。

【０１４２】

さらに、元データをネットワークを介して端末から受信し、この元データに対

して元部分データ、乱数部分データおよび分割部分データの生成処理を施して生成された複数の分割部分データをネットワークを介して保管サーバに送信して保管するので、多数のユーザが端末からネットワークを介してアクセスして分割処理を依頼することができ、共通化および経済化を図ることができる。

【０１４３】

【発明の効果】

以上説明したように、本発明によれば、元データを処理単位ビット長毎に区分けして複数の元部分データを生成し、複数の乱数部分データを生成し、各分割データを構成する各分割部分データを元部分データと乱数部分データの排他的論理和からなる所定の定義式に従って生成するので、従来のように多項式や剰余演算を用いることなく、コンピュータ処理に適したビット演算である排他的論理和演算を用いることにより高速かつ高性能な演算処理能力を必要とせず、大容量のデータに対しても簡単な演算処理を繰り返して分割データを簡単かつ迅速に生成することができるとともに、また分割データの保管に必要となる記憶容量も分割数に比例した倍数の容量よりも小さくすることができる。

【図面の簡単な説明】

【図１】

本発明の一実施形態に係るデータ分割方法を実施するデータ分割装置を含むシステム構成図である。

【図２】

図１に示す実施形態のデータ分割装置の分割数n=3の場合の分割処理を示すフローチャートである。

【図３】

１６ビットの元データSを８ビットの処理単位ビット長に基づいて分割数n=3で３分割する場合の各データと定義式および各分割部分データから元データを復元する場合の計算式などを示す表である。

【図４】

分割数n=3の場合の分割データ、分割部分データ、各分割部分データを生成する定義式を示す表である。

【図５】

図１に示す実施形態のデータ分割装置の分割数n=4の場合の分割処理を示すフローチャートである。

【図６】

分割数n=4の場合の分割データ、分割部分データ、各分割部分データを生成する定義式を示す表である。

【図７】

分割数n=5の場合の分割データ、分割部分データ、各分割部分データを生成する定義式を示す表である。

【図８】

図１に示す実施形態のデータ分割装置の分割数がnで処理単位ビット長がbである場合の一般的な分割処理を示すフローチャートである。

【図９】

図１に示す実施形態のデータ分割装置の分割数が２である場合の分割処理を示すフローチャートである。

【符号の説明】

１　分割装置

３　ネットワーク

５　端末

７ａ，７ｂ，７ｃ　保管サーバ

１１　分割データ生成手段

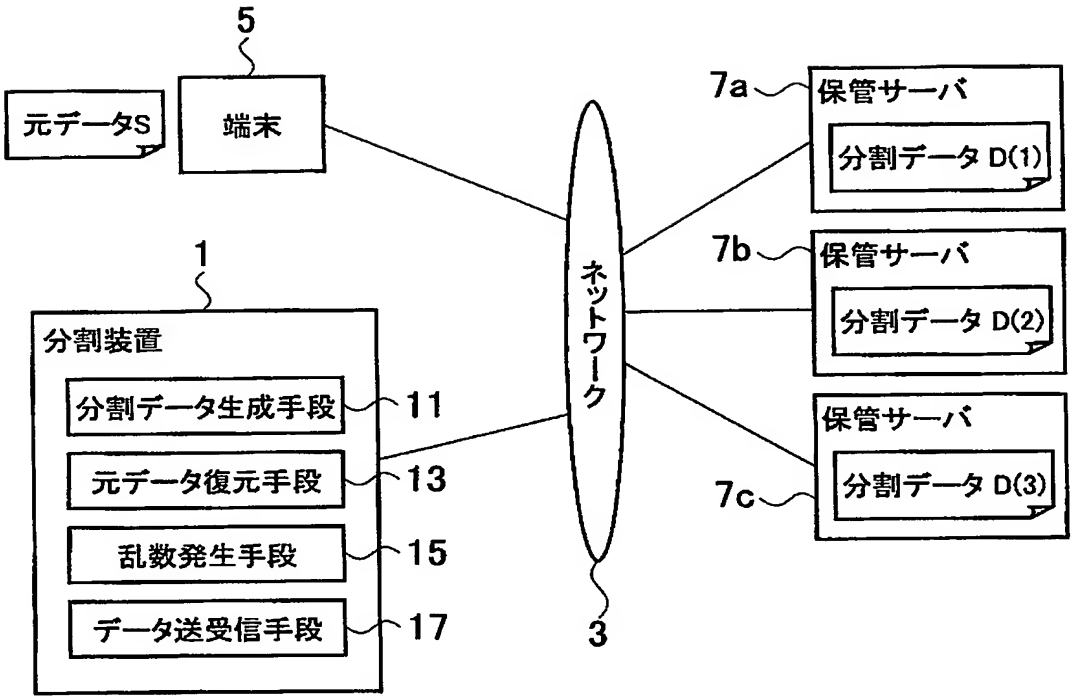
１３　元データ復元手段

１５　乱数発生手段

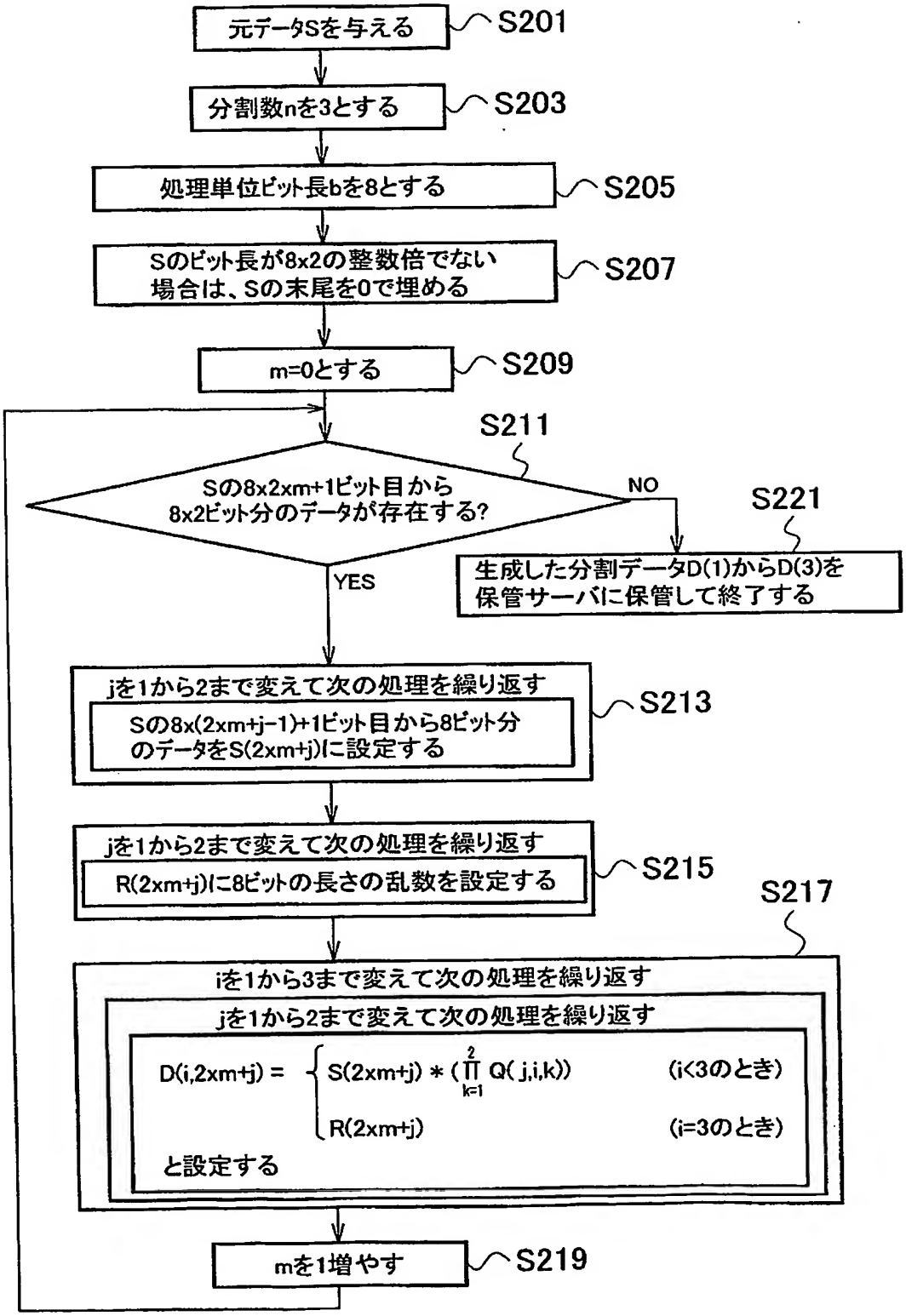
１７　データ送受信手段

【書類名】　図面

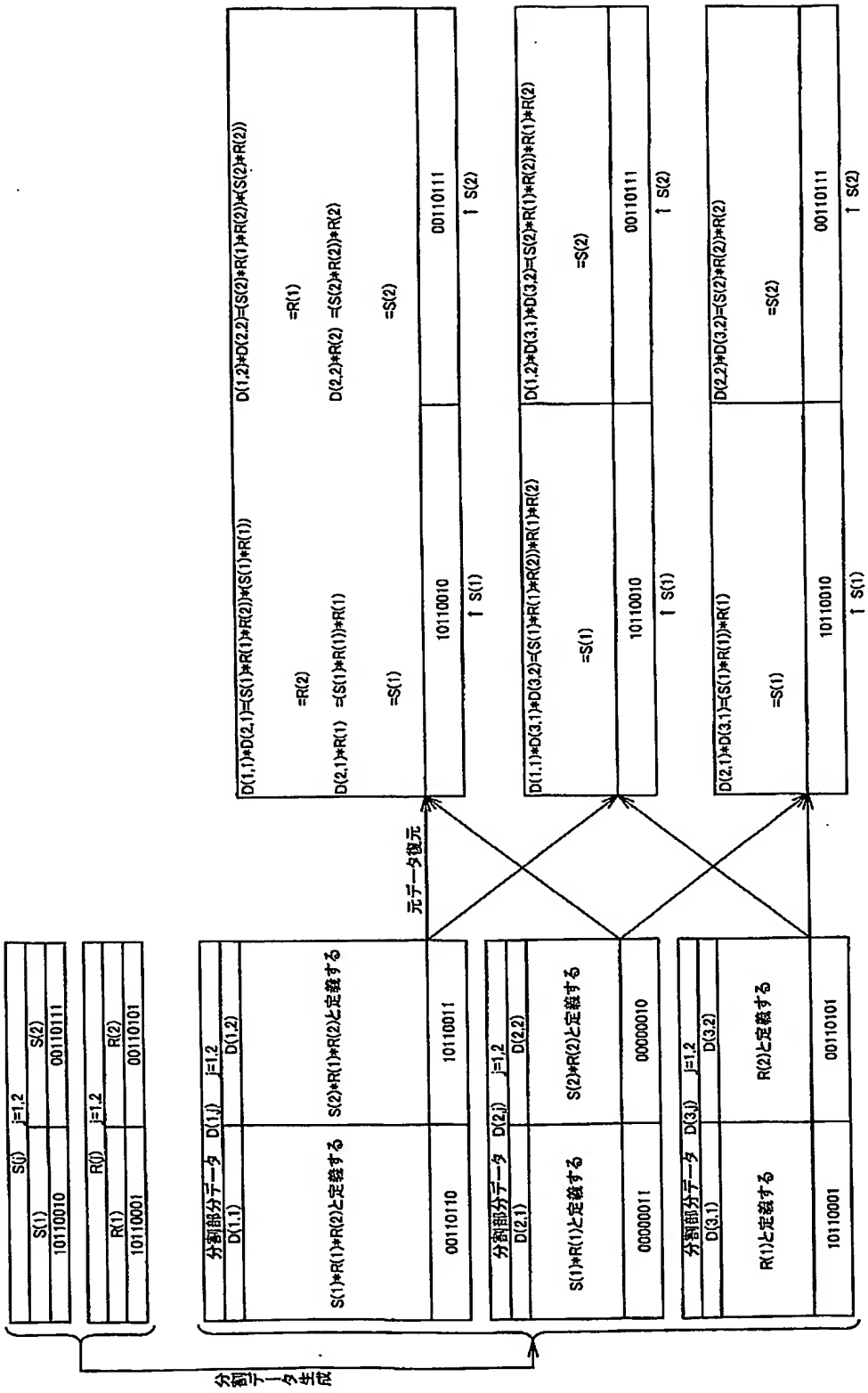
【図１】


5
元データS
端末
1
分割装置
分割データ生成手段
11
元データ復元手段
13
乱数発生手段
15
データ送受信手段
17
ネットワーク
3
7a
保管サーバ
分割データ D(1)
7b
保管サーバ
分割データ D(2)
保管サーバ
7c
分割データ D(3)

【図２】


元データSを与える
S201
分割数nを3とする
S203
処理単位ビット長bを8とする
S205
Sのビット長が8x2の整数倍でない
場合は、Sの末尾を0で埋める
S207
m=0とする
S209
S211
Sの8x2xm+1ビット目から
8x2ビット分のデータが存在する?
NO
S221
生成した分割データD(1)からD(3)を
保管サーバに保管して終了する
YES
jを1から2まで変えて次の処理を繰り返す
Sの8x(2xm+j-1)+1ビット目から8ビット分
のデータをS(2xm+j)に設定する
S213
jを1から2まで変えて次の処理を繰り返す
R(2xm+j)に8ビットの長さの乱数を設定する
S215
S217
iを1から3まで変えて次の処理を繰り返す
jを1から2まで変えて次の処理を繰り返す
D(i,2xm+j) = S(2xm+j) * (∏ k=1 2 Q(j,i,k)) (i<3のとき)
R(2xm+j) (i=3のとき)
と設定する
mを1増やす
S219

【図３】


S(j) j=1,2
S(1)
10110010
S(2)
00110111
R(j) j=1,2
R(1)
10110001
R(2)
00110101
分割データ生成
分割部分データ D(1,j) j=1,2
D(1,1)
S(1)*R(1)+R(2)と定義する
D(1,2)
S(2)*R(1)+R(2)と定義する
00110110
10110011
分割部分データ D(2,j) j=1,2
D(2,1)
S(1)*R(1)と定義する
D(2,2)
S(2)*R(2)と定義する
00000011
00000010
分割部分データ D(3,j) j=1,2
D(3,1)
R(1)と定義する
D(3,2)
R(2)と定義する
10110001
00110101
元データ復元
D(1,1)*D(2,1)=(S(1)*R(1)+R(2))*(S(1)*R(1))
=R(2)
D(2,1)*R(1) =(S(1)*R(1))*R(1)
=S(1)
D(1,2)*D(2,2)=(S(2)*R(1)+R(2))*(S(2)*R(2))
=R(1)
D(2,2)*R(2) =(S(2)*R(2))*R(2)
=S(2)
10110010
00110111
↑S(1)
↑S(2)
D(1,1)*D(3,1)*D(3,2)=(S(1)*R(1)+R(2))*R(1)*R(2)
=S(1)
D(1,2)*D(3,1)*D(3,2)=(S(2)*R(1)+R(2))*R(1)*R(2)
=S(2)
10110010
00110111
↑S(1)
↑S(2)
D(2,1)*D(3,1)=(S(1)*R(1))*R(1)
=S(1)
D(2,2)*D(3,2)=(S(2)*R(2))*R(2)
=S(2)
10110010
00110111
↑S(1)
↑S(2)

【図４】

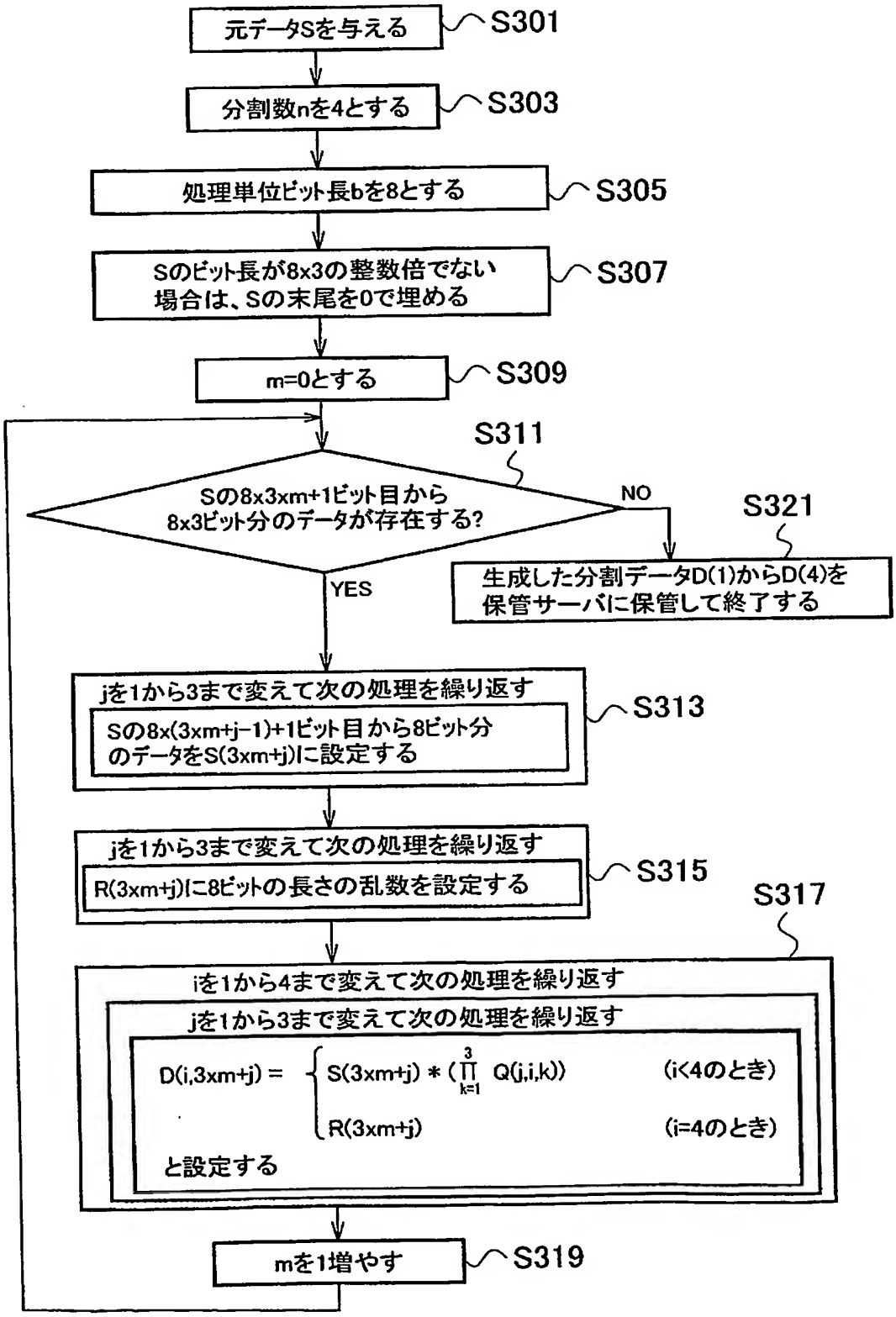
3分割 (n=3)

任意の2つの分割データから元データが復元可能。

(mはm>0の任意の整数)　　→元データSの末尾まで続く

jの値	1	2	…	j=2×m+1	j+1	…
元データ　S(j)	S(1)	S(2)	…	S(j)	S(j+1)	…
乱数　　R(j)	R(1)	R(2)	…	R(j)	R(j+1)	…
分割部分データD(1,j)	S(1)*R(1)*R(2)	S(2)*R(1)*R(2)	…	S(j)*R(j)*R(j+1)	S(j+1)*R(j)*R(j+1)	…
分割部分データD(2,j)	S(1)*R(1)	S(2)　*R(2)	…	S(j)*R(j)	S(j+1)　*R(j+1)	…
分割部分データD(3,j)	R(1)	R(2)	…	R(j)	R(j+1)	…

【図５】


元データSを与える
S301
分割数nを4とする
S303
処理単位ビット長bを8とする
S305
Sのビット長が8x3の整数倍でない場合は、Sの末尾を0で埋める
S307
m=0とする
S309
S311
Sの8x3xm+1ビット目から8x3ビット分のデータが存在する?
NO
S321
生成した分割データD(1)からD(4)を保管サーバに保管して終了する
YES
jを1から3まで変えて次の処理を繰り返す
Sの8x(3xm+j-1)+1ビット目から8ビット分のデータをS(3xm+j)に設定する
S313
jを1から3まで変えて次の処理を繰り返す
R(3xm+j)に8ビットの長さの乱数を設定する
S315
S317
iを1から4まで変えて次の処理を繰り返す
jを1から3まで変えて次の処理を繰り返す
D(i,3xm+j) = { S(3xm+j) * (∏ k=1 3 Q(j,i,k)) (i<4のとき)
R(3xm+j) (i=4のとき)
と設定する
mを1増やす
S319

【図６】

4分割 (n=4)

任意の3つの分割データ(取り方によっては2つの分割データ)から元データが復元可能。

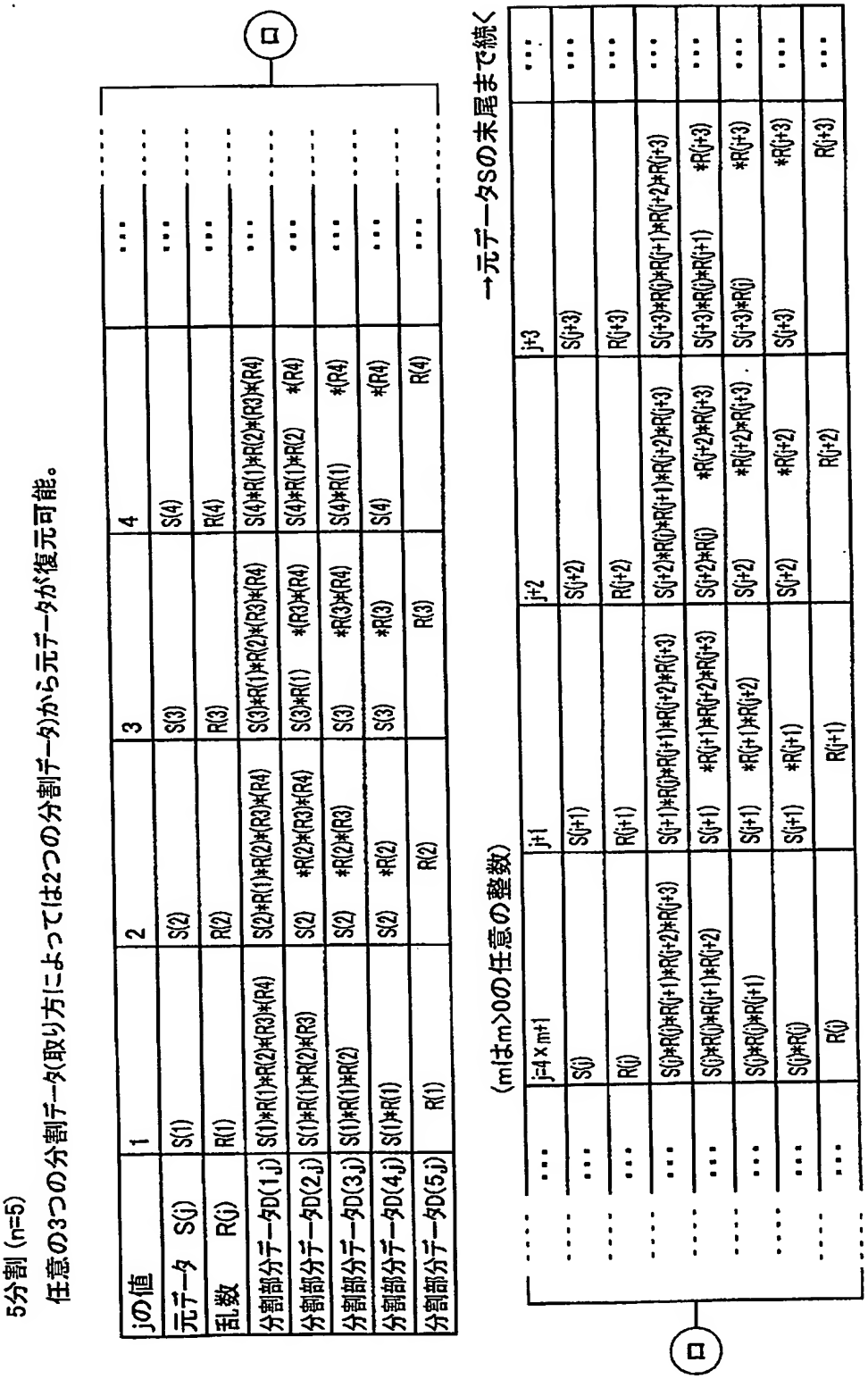
jの値	1	2	3	...
元データ　S(j)	S(1)	S(2)	S(3)	...
乱数　　R(j)	R(1)	R(2)	R(3)	...
分割部分データD(1,j)	S(1)*R(1)*R(2)*(R3)	S(2)*R(1)*R(2)*(R3)	S(3)*R(1)*R(2)*(R3	...
分割部分データD(2,j)	S(1)*R(1)*R(2)	S(2)　*R(2)*(R3)	S(3)*R(1)　*(R3)	...
分割部分データD(3,j)	S(1)*R(1)	S(2)　*R(2)	S(3)　　*R(3)	...
分割部分データD(4,j)	R(1)	R(2)	R(3)	...

(mはm>0の任意の整数)

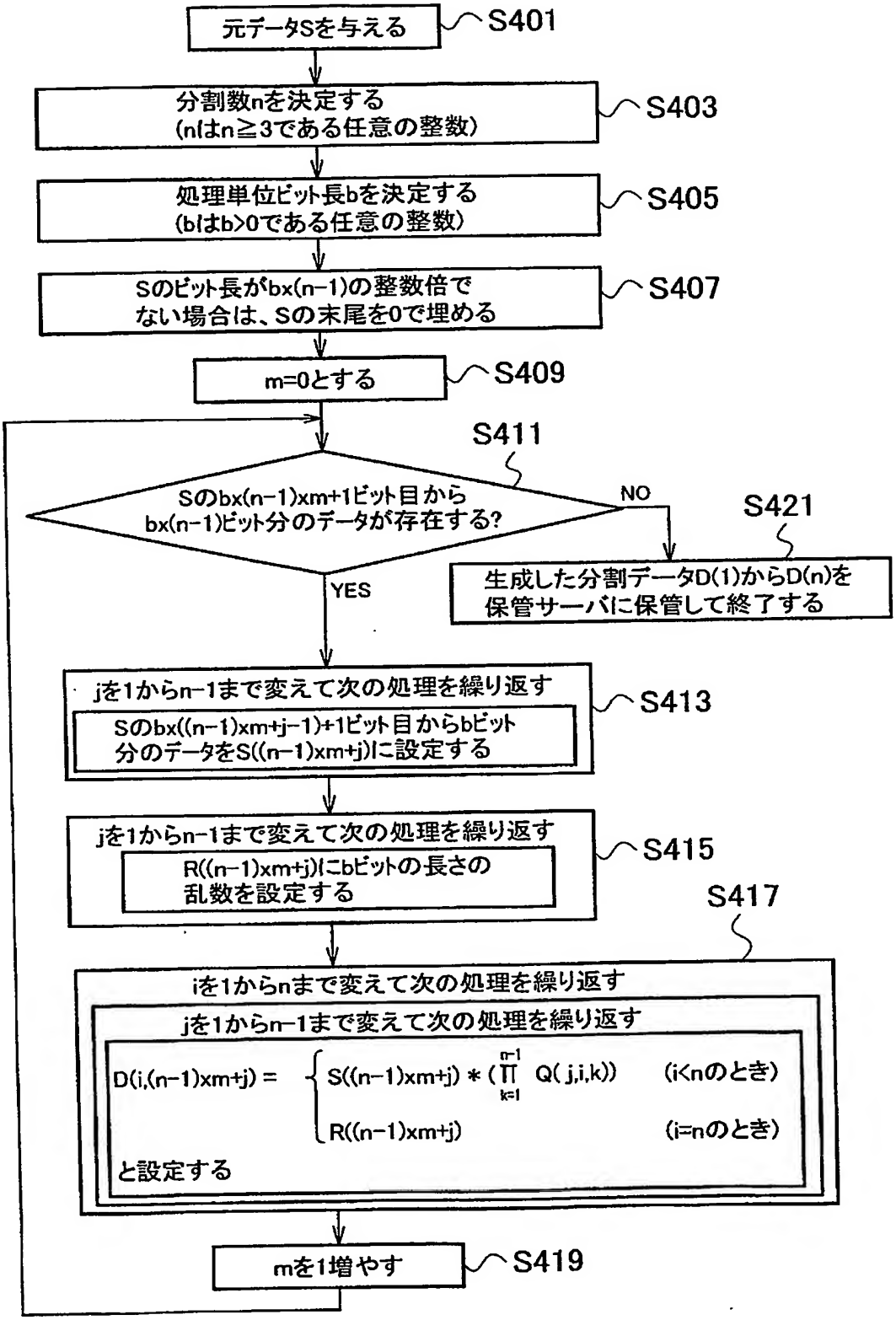
...	j=3 x m+1	j+1	j+2	...
...	S(j)	S(j+1)	S(j+2)	...
...	R(j)	R(j+1)	R(j+2)	...
...	S(j)*R(j)*R(j+1)*R(j+2)	S(j+1)*R(j)*R(j+1)*R(j+2)	S(j+2)*R(j)*R(j+1)*R(j+2)	...
...	S(j)*R(j)*R(j+1)	S(j+1)　*R(j+1)*R(j+2)	S(j+2)*R(j)　*R(j+2)	...
...	S(j)*R(j)	S(j+1)　*R(j+1)	S(j+2)　　*R(j+2)	...
...	R(j)	R(j+1)	R(j+2)	...

→元データSの末尾まで続く

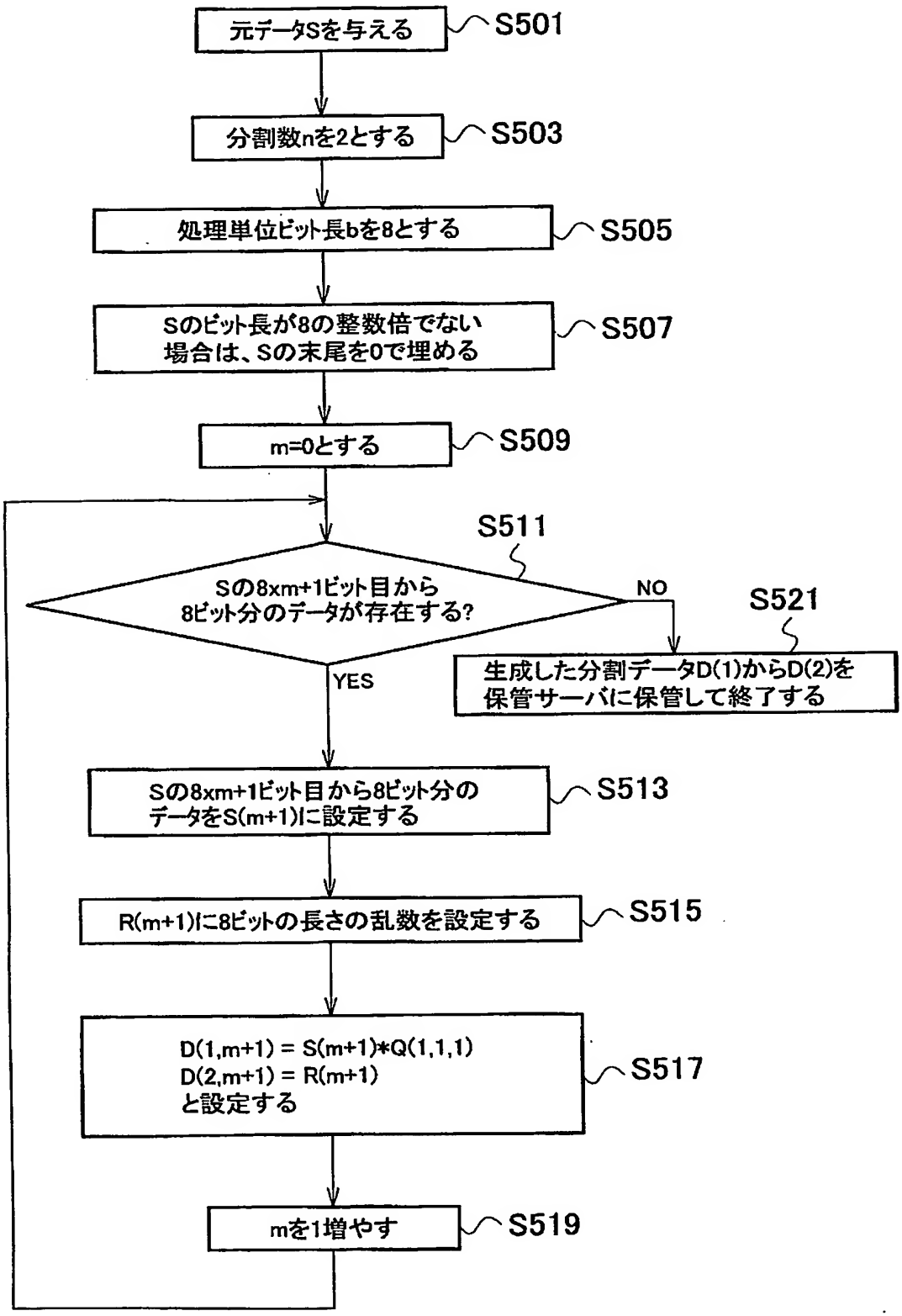
【図７】


5分割 (n=5)
任意の3つの分割データ(取り方によっては2つの分割データ)から元データが復元可能。
jの値
元データ S(j)
乱数 R(j)
分割部分データD(1,j)
分割部分データD(2,j)
分割部分データD(3,j)
分割部分データD(4,j)
分割部分データD(5,j)
(mは m>0の任意の整数)
j=4×m+1
j+1
j+2
j+3
→元データSの末尾まで続く

【図８】


元データSを与える
S401
分割数nを決定する
(nはn≧3である任意の整数)
S403
処理単位ビット長bを決定する
(bはb>0である任意の整数)
S405
Sのビット長がbx(n−1)の整数倍で
ない場合は、Sの末尾を0で埋める
S407
m=0とする
S409
S411
Sのbx(n−1)xm+1ビット目から
bx(n−1)ビット分のデータが存在する?
NO
S421
YES
生成した分割データD(1)からD(n)を
保管サーバに保管して終了する
jを1からn−1まで変えて次の処理を繰り返す
Sのbx((n−1)xm+j−1)+1ビット目からbビット
分のデータをS((n−1)xm+j)に設定する
S413
jを1からn−1まで変えて次の処理を繰り返す
R((n−1)xm+j)にbビットの長さの
乱数を設定する
S415
S417
iを1からnまで変えて次の処理を繰り返す
jを1からn−1まで変えて次の処理を繰り返す
D(i,(n−1)xm+j) = { S((n−1)xm+j) * (∏ k=1 n−1 Q(j,i,k)) (i<nのとき)
R((n−1)xm+j) (i=nのとき)
と設定する
mを1増やす
S419

【図９】


元データSを与える
S501
分割数nを2とする
S503
処理単位ビット長bを8とする
S505
Sのビット長が8の整数倍でない
場合は、Sの末尾を0で埋める
S507
m=0とする
S509
S511
Sの8xm+1ビット目から
8ビット分のデータが存在する?
NO
S521
生成した分割データD(1)からD(2)を
保管サーバに保管して終了する
YES
Sの8xm+1ビット目から8ビット分の
データをS(m+1)に設定する
S513
R(m+1)に8ビットの長さの乱数を設定する
S515
D(1,m+1) = S(m+1)*Q(1,1,1)
D(2,m+1) = R(m+1)
と設定する
S517
mを1増やす
S519

【書類名】　要約書

【要約】

【課題】　比較的簡単な処理により元データを効率的に分割し得るデータ分割方法および装置を提供する。

【解決手段】　元データS、分割数n、処理単位ビット長bを設定し（ステップＳ２０１－Ｓ２０５）、元データSを処理単位ビット長b毎に区分けして複数の元部分データS(j)を生成し（ステップＳ２１３）、複数の乱数部分データR(j)を生成し（ステップＳ２１５）、各分割データD(i)を構成する各分割部分データ(i,j)を元部分データと乱数部分データの排他的論理和からなる所定の定義式に従って生成する（ステップＳ２１７）。

【選択図】　図１

特願２００２－３６７６０８

出願人履歴情報

識別番号　[３９９０３５７６６]

１．変更年月日　１９９９年　６月　９日
[変更理由]　新規登録
住　所　東京都千代田区内幸町一丁目１番６号
氏　名　エヌ・ティ・ティ・コミュニケーションズ株式会社